

Sanguthevar Rajasekaran (Ed.)

LNBI 5462

Bioinformatics and Computational Biology

First International Conference, BICoB 2009
New Orleans, LA, USA, April 2009
Proceedings



 Springer

The Springer logo, which consists of a stylized white chess knight (horse) facing left, positioned above the word "Springer" in a white serif font.

Lecture Notes in Bioinformatics

5462

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

Editorial Board: A. Apostolico S. Brunak M. Gelfand
T. Lengauer S. Miyano G. Myers M.-F. Sagot D. Sankoff
R. Shamir T. Speed M. Vingron W. Wong

Subseries of Lecture Notes in Computer Science

Sanguthevar Rajasekaran (Ed.)

Bioinformatics and Computational Biology

First International Conference, BICoB 2009
New Orleans, LA, USA, April 8-10, 2009
Proceedings



Springer

Series Editors

Sorin Istrail, Brown University, Providence, RI, USA

Pavel Pevzner, University of California, San Diego, CA, USA

Michael Waterman, University of Southern California, Los Angeles, CA, USA

Volume Editor

Sanguthevar Rajasekaran

University of Connecticut, Department of Computer Science and Engineering

257 ITE Building, 371 Fairfield Way, Storrs, CT 06269-2155, USA

E-mail: rajasek@engr.uconn.edu

Library of Congress Control Number: Applied for

CR Subject Classification (1998): H.2.8, J.3, I.5, I.2, H.3, F.1-2

LNCS Sublibrary: SL 8 – Bioinformatics

ISSN 0302-9743

ISBN-10 3-642-00726-0 Springer Berlin Heidelberg New York

ISBN-13 978-3-642-00726-2 Springer Berlin Heidelberg New York

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

springer.com

© Springer-Verlag Berlin Heidelberg 2009

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper SPIN: 12633326 06/3180 5 4 3 2 1 0

Preface

This volume presents the proceedings of the First International Conference on Bioinformatics and Computational Biology (BICoB 2009). This conference was supported by the International Society for Computers and Applications (ISCA) and Springer.

Computational techniques have already enabled unprecedented advances in modern biology and medicine. This continues to be a vibrant research area with broadening of computational techniques and new emerging challenges. The Bioinformatics and Computational Biology (BICoB) conference has the goal of promoting the advancement of computing techniques and their application to life sciences. The topics of interest include (and are not limited to):

- Genome analysis: genome assembly; genome and chromosome annotation, gene finding; alternative splicing; EST analysis and comparative genomics
- Sequence analysis: multiple sequence alignment; sequence search and clustering; function prediction, motif discovery, functional site recognition in protein, RNA and DNA sequences
- Phylogenetics: phylogeny estimation; models of evolution; comparative biological methods; population genetics
- Structural Bioinformatics: structure matching, prediction, analysis and comparison; methods and tools for docking; protein design
- Analysis of high-throughput biological data: microarrays (nucleic acid, protein, array CGH, genome tiling, and other arrays); EST; SAGE; MPSS; proteomics; mass spectrometry
- Genetics and population analysis: linkage analysis; association analysis; population simulation; haplotyping; marker discovery; genotype calling
- Systems biology: systems approaches to molecular biology; multiscale modeling; pathways; gene networks

BICoB is interested in all areas of computing with an impact on life sciences including (but not limited to) algorithms, databases, languages, systems, and high-performance computing. Examples include:

- Parallel and high-performance techniques
- Unifying computational techniques
- Data and image mining techniques
- Approximation and randomized algorithms and systems
- Computational biology on emerging architectures and hardware accelerators

In BICoB 2009 there were three keynote speeches, ten invited talks, and 30 contributed presentations (selected from a total of 72 submissions). We are grateful to the keynote speakers and the invited speakers who contributed tremendously to the success of the conference. We are also thankful to all the authors who submitted papers, especially those who gave presentations at the conference.

The Program Committee members as well as reviewers recruited by them deserve a special thanks for their meticulous work in selecting the contributed papers. We are also grateful to Sahar Al Seesi for finalizing the papers for the proceedings.

April 2009

Sanguthevar Rajasekaran
Srinivas Aluru
Limsoon Wang

Organization

The First International Conference on BioInformatics and Computational Biology (BICoB) was organized by the International Society of Computers and Applications (ISCA) in collaboration with Springer

Executive Committee

General Chair	Sanguthevar Rajasekaran (University of Connecticut)
Program Co-chairs	Srinivas Aluru (Iowa State University) Limsoon Wang (National University of Singapore)
Steering Committee	Reda Ammar (University of Connecticut) Tao Jiang (University of California, Riverside) Vipin Kumar (University of Minnesota) Ming Li (University of Waterloo) Sanguthevar Rajasekaran, Chair (University of Connecticut) John Reif (Duke University) Sartaj Sahni (University of Florida, Gainesville)
Publicity Co-chairs	Ian Greenshields (University of Connecticut) Chun-Hsi Huang (University of Connecticut)
Keynote Speakers	Richard Karp, University of California, Berkeley Ming Li, University of Waterloo Vineet Bafna, University of California, San Diego
Invited Speakers	Srinivas Aluru, Iowa State University Vladimir Filkov, University of California, Davis Vipin Kumar, University of Minnesota Bin Ma, University of Waterloo Ion Mandoiu, University of Connecticut Satoru Miyano, University of Tokyo T.M. Murali, Virginia Tech. Mona Singh, Princeton University Wing-Kin Sung, National University of Singapore Dong Xu, University of Missouri, Columbia
Proceedings Coordinator	Sahar Al Seesi, University of Connecticut

Program Committee

Richa Agarwala	National Institute of Health
Yutaka Akiyama	Tokyo Institute of Technology
Ziv Bar-Joseph	Carnegie Mellon University
Chiranjib Bhattacharyya	Indian Institute of Science
Kun-Mao Chao	National Taiwan University
Jake Chen	Indiana University
Francis Y.L. Chin	Hong Kong University
Bhaskar Dasgupta	University of Illinois, Chicago
Scott J. Emrich	University of Notre Dame
Oliver Eulenstein	Iowa State University
Vladimir Filkov	University of California, Davis
Dmitrij Frishman	Tech. University of Munich
Terry Gaasterland	University of California, San Diego
Gaston Gonnet	ETH, Zurich
Osamu Gotoh	Kyoto University
Wen-Lian Hsu	Academia Sinica
Ming-Yang Kao	Northwestern University
Marek Karpinski	University of Bonn
George Karypis	University of Minnesota
Danny Krizanc	Wesleyan University
Bin Ma	University of Waterloo
Ion Mandoiu	University of Connecticut
Shinichi Morishita	University of Tokyo
Giri Narasimhan	Florida International University
Enno Ohlebusch	University of Ulm
Yi Pan	Georgia State University
Srinivasan Parthasarathy	Ohio State University
Prabakaran Ponraj	Duke University
Mihai Pop	University of Maryland
David Posada	University of Vigo, Spain
Ben Raphael	Brown University
Knut Reinert	Freie University of Berlin
Isidore Rigoutsos	IBM TJ Watson Research Center
Marie-France Sagot	INRIA Rhone-Alpes
Mona Singh	Princeton University
Wing-Kin Sung	National University of Singapore
Sing-Hoi Sze	Texas A&M University
Jerzy Tiuryn	Warsaw University
Ugo Vaccaro	University of Salerno
Li-San Wang	University of Pennsylvania
Yufeng Wu	University of Connecticut
Dong Xu	University of Missouri, Columbia
Alex Zelikovskiy	Georgia State University

Reviewers

M. Bader	J. He	U. Schöning
S. Balla	A. Kolinski	O. Schulztrieglaff
M. Bauer	T.-H. Lin	S. Soi
P. Biecek	Y. Lu	M. Song
K. Cao	R. Preisner	E. Szczurek
J. Davila	Y. Qi	D. Ucar
J. Ernst	S. Roy	B. Wilczynski
A. Fox	P. Ryvkin	

Sponsoring Institutions

University of Connecticut, Storrs

Booth Engineering Center for Advanced Technologies, Storrs

Table of Contents

Invited Talks

Association Analysis Techniques for Bioinformatics Problems	1
<i>Gowtham Atluri, Rohit Gupta, Gang Fang, Gaurav Pandey, Michael Steinbach, and Vipin Kumar</i>	
Analyzing and Interrogating Biological Networks (Abstract)	14
<i>Eric Banks, Elena Nabieva, Bernard Chazelle, Ryan Peterson, and Mona Singh</i>	
From Architecture to Function (and Back) in Bio-networks	16
<i>Vladimir Filkov</i>	
A New Machine Learning Approach for Protein Phosphorylation Site Prediction in Plants	18
<i>Jianjiong Gao, Ganesh Kumar Agrawal, Jay J. Thelen, Zoran Obradovic, A. Keith Dunker, and Dong Xu</i>	
Assembly of Large Genomes from Paired Short Reads	30
<i>Benjamin G. Jackson, Patrick S. Schnable, and Srinivas Aluru</i>	
Amino Acid Classification and Hash Seeds for Homology Search	44
<i>Weiming Li, Bin Ma, and Kaizhong Zhang</i>	
Genotype and Haplotype Reconstruction from Low-Coverage Short Sequencing Reads	52
<i>Ion Măndoiu</i>	
Gene Networks Viewed through Two Models	54
<i>Satoru Miyano, Rui Yamaguchi, Yoshinori Tamada, Masao Nagasaki, and Seiya Imoto</i>	
Identifying Evolutionarily Conserved Protein Interaction Modules Using GraphHopper	67
<i>Corban G. Rivera and T.M. Murali</i>	
The 2-Interval Pattern Matching Problems and Its Application to ncRNA Scanning	79
<i>Thomas K.F. Wong, S.M. Yiu, T.W. Lam, and Wing-Kin Sung</i>	

Refereed Papers

RNA Pseudoknot Folding through Inference and Identification Using TAG _{RNA}	90
<i>Sahar Al Seesi, Sanguthevar Rajasekaran, and Reda Ammar</i>	

Comparing Bacterial Genomes by Searching Their Common Intervals	102
<i>Sébastien Angibaud, Damien Eveillard, Guillaume Fertin, and Irena Rusu</i>	
Generalized Binary Tanglegrams: Algorithms and Applications	114
<i>Mukul S. Bansal, Wen-Chieh Chang, Oliver Eulenstein, and David Fernández-Baca</i>	
Three-Dimensional Multimodality Modelling by Integration of High-Resolution Interindividual Atlases and Functional MALDI-IMS Data	126
<i>Felix Bollenbeck, Stephanie Kaspar, Hans-Peter Mock, Diana Weier, and Udo Seiffert</i>	
Detecting Motifs in a Large Data Set: Applying Probabilistic Insights to Motif Finding	139
<i>Christina Boucher and Daniel G. Brown</i>	
A Biclustering Method to Discover Co-regulated Genes Using Diverse Gene Expression Datasets	151
<i>Doruk Bozdağ, Jeffrey D. Parvin, and Umit V. Catalyurek</i>	
Computational Protocol for Screening GPI-anchored Proteins	164
<i>Wei Cao, Kazuya Sumikoshi, Tohru Terada, Shugo Nakamura, Katsuhiko Kitamoto, and Kentaro Shimizu</i>	
Towards Large-Scale Molecular Dynamics Simulations on Graphics Processors	176
<i>Joseph E. Davis, Adnan Ozsoy, Sandeep Patel, and Michela Taufer</i>	
An Agent-Based Model of Solid Tumor Progression	187
<i>Didier Dréau, Dimitre Stanimirov, Ted Carmichael, and Mirsad Hadzikadic</i>	
Deciphering Drug Action and Escape Pathways: An Example on Nasopharyngeal Carcinoma	199
<i>Difeng Dong, Chun-Ying Cui, Benjamin Mow, and Limsoon Wong</i>	
Selection of Multiple SNPs in Case-Control Association Study Using a Discretized Network Flow Approach	211
<i>Shantanu Dutt, Yang Dai, Huan Ren, and Joel Fontanarosa</i>	
Biclustering Expression Data Based on Expanding Localized Substructures	224
<i>Cesim Erten and Melih Sözdinler</i>	
Constrained Fisher Scores Derived from Interaction Profile Hidden Markov Models Improve Protein to Protein Interaction Prediction	236
<i>Alvaro J. González and Li Liao</i>	

Improving Protein Localization Prediction Using Amino Acid Group Based Physichemical Encoding	248
<i>Jianjun Hu and Fan Zhang</i>	
The Impact of Gene Selection on Imbalanced Microarray Expression Data	259
<i>Abu H.M. Kamal, Xingquan Zhu, Abhijit S. Pandya, Sam Hsu, and Muhammad Shoaib</i>	
Spatial Information and Boolean Genetic Regulatory Networks	270
<i>Matthieu Manceny, Marc Aiguier, Pascale Le Gall, Joan Hérisson, Ivan Junier, and François Képès</i>	
Modeling of Genetic Regulatory Network in Stochastic π -Calculus.....	282
<i>Myène Maurin, Morgan Magnin, and Olivier Roux</i>	
fMRI Activation Detection by MultiScale Hidden Markov Model	295
<i>Fangyuan Nan, Yaonan Wang, and Xiaoping Ma</i>	
cnF2freq: Efficient Determination of Genotype and Haplotype Probabilities in Outbred Populations Using Markov Models	307
<i>Carl Nettelblad, Sverker Holmgren, Lucy Crooks, and Örjan Carlborg</i>	
A Comprehensive Analysis Workflow for Genome-Wide Screening Data from ChIP-Sequencing Experiments	320
<i>Hatice Gulcin Ozer, Doruk Bozdağ, Terry Camerlengo, Jiejun Wu, Yi-Wen Huang, Tim Hartley, Jeffrey D. Parvin, Tim Huang, Umit V. Catalyurek, and Kun Huang</i>	
A Fitness Distance Correlation Measure for Evolutionary Trees	331
<i>Hyun Jung Park and Tiffani L. Williams</i>	
Alignment and Analysis of Closely Related Genomes	343
<i>Allison Regier, Michael Olson, and Scott J. Emrich</i>	
Computational Prediction of Genes Translationally Regulated by Cytoplasmic Polyadenylation Elements	353
<i>Eric C. Rouchka, Xiangping Wang, James H. Graham, and Nigel G.F. Cooper</i>	
Multiple Sequence Alignment System for Pyrosequencing Reads.....	362
<i>Fahad Saeed, Ashfaq Khokhar, Osvaldo Zagordi, and Niko Beerenwinkel</i>	
A Bayesian Approach to High-Throughput Biological Model Generation	376
<i>Xinghua Shi and Rick Stevens</i>	

Parallel Selection of Informative Genes for Classification	388
<i>Michael Slavik, Xingquan Zhu, Imad Mahgoub, and Muhammad Shoaib</i>	
Simulation Methods in Uncovering New Regulatory Mechanisms in Signaling Pathways	400
<i>Jarostaw Smieja</i>	
GridSPiM: A Framework for Simple Locality and Containment in the Stochastic π -Calculus	409
<i>Stephen Tyree, Rayus Kuplicki, Trevor Sarratt, Scott Fujan, and John Hale</i>	
Mutual Information Based Extrinsic Similarity for Microarray Analysis	424
<i>Duygu Ucar, Fatih Altiparmak, Hakan Ferhatosmanoglu, and Srinivasan Parthasarathy</i>	
Graph Spectral Approach for Identifying Protein Domains	437
<i>Hari Krishna Yalamanchili and Nita Parekh</i>	
Author Index	449

Association Analysis Techniques for Bioinformatics Problems

Gowtham Atluri, Rohit Gupta, Gang Fang, Gaurav Pandey,
Michael Steinbach, and Vipin Kumar

Department of Computer Science and Engineering, University of Minnesota
{gowtham, rohit, gangfang, gaurav, steinbac, kumar}@cs.umn.edu
<http://www.cs.umn.edu/~kumar/dmbio>

Abstract. Association analysis is one of the most popular analysis paradigms in data mining. Despite the solid foundation of association analysis and its potential applications, this group of techniques is not as widely used as classification and clustering, especially in the domain of bioinformatics and computational biology. In this paper, we present different types of association patterns and discuss some of their applications in bioinformatics. We present a case study showing the usefulness of association analysis-based techniques for pre-processing protein interaction networks for the task of protein function prediction. Finally, we discuss some of the challenges that need to be addressed to make association analysis-based techniques more applicable for a number of interesting problems in bioinformatics.

Keywords: Data Mining, Association Analysis, Bioinformatics, Frequent Pattern Mining.

1 Introduction

The area of data mining known as association analysis¹ [1,2,50] seeks to find patterns that describe the relationships among the binary attributes (variables) used to characterize a set of objects. The iconic example of data sets analyzed by these techniques is market basket data, where the objects are transactions consisting of sets of items purchased by a customer, and the attributes are binary variables that indicate whether or not an item was purchased by a particular customer. The interesting patterns in these data sets are either sets of items that are frequently purchased together (frequent itemset patterns) or rules that capture the fact that the purchase of one set of items often implies the purchase of a second set of items (association rule patterns). Association patterns, whether rules or itemsets, are local patterns in that they hold only for a subset of transactions. The size of this set of supporting transactions, which is known as the support of the pattern, is one measure of the strength of a pattern. A key

¹ Not to be confused with the related, but separate field of statistical association analysis [3].

strength of association pattern mining is that the potentially exponential nature of the search can often be made tractable by using support based pruning of patterns [1], i.e., the elimination of patterns supported by too few transactions early on in the search process. Efforts to date have created a well-developed conceptual (theoretical) foundation [64] and an efficient set of algorithms [2,20]. The framework has been extended well beyond the original application to market basket data to encompass new applications [8,24,23,57].

Despite the solid foundations of association analysis and the potential economic and intellectual benefits of pattern discovery and its various applications, this group of techniques is not widely used as a data analysis tool in bioinformatics and computational biology. Some prominent examples of these data types are gene expression data [33] and data on genetic variations (e.g., single nucleotide polymorphism (SNP) data) [22]. Although the use of clustering and classification techniques is common for the analysis of these and other biological data sets, techniques from association analysis are rarely employed (The few exceptions include the work of researchers [5,13,30,29,40], including ourselves [57,37,35]). For instance, for the problem of protein function prediction, which is a key problem in bioinformatics [52], recent surveys [36,48,17] discuss several hundred papers using clustering and classification techniques, but only a handful using association analysis techniques. Thus, it has to be acknowledged that association analysis techniques have not found widespread use in this important domain.

In this paper we discuss some applications of association analysis techniques in bioinformatics and the challenges that need to be addressed to make these techniques applicable to other problems in this promising area. The rest of the paper is organized as follows: Section 2 presents a brief overview of various types of association patterns, which can be very useful for discovering different forms of knowledge from complex data sets, such as those generated by high-throughput biological studies. In the next section, we discuss a case study of how an association measure, h - confidence, can be used to address issues with the quality of the currently available protein interaction data. Section 3 discusses the use of association patterns for a bioinformatics application, namely addressing the noise and incompleteness issues with the currently available protein interaction network data. Section 4 provides concluding remarks and some of the challenges that needs to be addressed to extend the application of association patterns to a wide range of problems in bioinformatics.

2 Association Patterns

This section introduces some commonly used association patterns that have been proposed in the literature.

2.1 Traditional Frequent Patterns

Traditional frequent pattern analysis [50] focuses on binary transaction data, such as the data that results when customers purchase items in, for example, a grocery

store. Such market basket data can be represented as a collection of transactions, where each transaction corresponds to the items purchased by a specific customer. More formally, data sets of this type can be represented as a binary matrix, where there is one row for each transaction, one column for each item, and the ij^{th} entry is 1 if the i^{th} customer purchased the j^{th} item, and 0 otherwise.

Given such a binary matrix representation, a key task in association analysis is to finding frequent itemsets in this matrix, which are sets of items that frequently occur together in a transaction. The strength of an itemset is measured by its *support*, which is the number (or fraction) of transactions in the data set in which all items of the itemset appear together. Interestingly, support is an anti-monotonic measure in that the support of an itemset in a given data set can not be less than any of its supersets. This anti-monotonicity property allows the design of several efficient algorithms, such as Apriori [2] and FPGrowth [20], for discovering frequent itemsets in a given binary data matrix. However, an important factor in choosing the threshold for the minimum support of an itemset to be considered frequent is computational efficiency. Specifically, if n is the number of binary attributes in a transaction data set, there are potentially $2^n - 1$ possible non-empty itemsets. Since transaction data is typically sparse, i.e., contains mostly 0's, the number of frequent itemsets is far less than $2^n - 1$. However, the actual number depends greatly on the support threshold that is chosen. Nonetheless, with judicious choices for the support threshold, the number of patterns discovered from a data set can be made manageable. Also, note that, in addition to support, a number of additional measures have been proposed to determine the interestingness of association patterns [49].

2.2 Hyperclique Patterns

A hyperclique pattern [61] is a type of frequent pattern that contains items that are strongly associated with each other over the supporting transaction, and are quite sparse (mostly 0) over the rest of the transactions. As discussed above, in

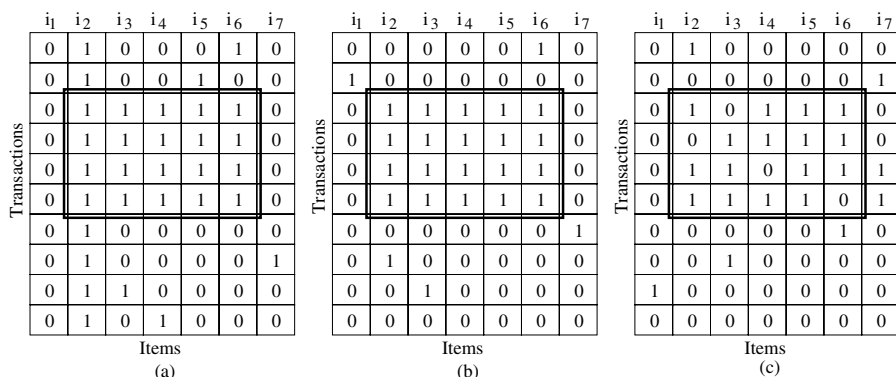


Fig. 1. Different types of association patterns (a) Traditional Frequent Patterns (b) Hyperclique Patterns (c) Error-tolerant Patterns

traditional frequent pattern mining, choosing the right support threshold can be quite tricky. If support threshold is too high, we may miss many interesting patterns involving low support items. If support is too low, it becomes difficult to mine all the frequent patterns because the number of extracted patterns increases substantially, many of which may relate a high-frequency item to a low-frequency item. Such patterns, which are called *cross-support patterns*, are likely to be spurious. For example, the pattern in Figure 1(a), $\{i_2, i_3, i_4, i_5, i_6\}$ includes a high-frequency item i_2 , which does not appear to have any specific association with other items in the patterns. Hyperclique patterns avoid these cross-support patterns by defining an anti-monotonic association measure known as *h-confidence* that ensures a high affinity between the itemsets constituting a hyperclique pattern [61]. Formally, the h-confidence of an itemset $X = \{i_1, i_2, \dots, i_m\}$, denoted as $hconf(X)$, is defined as,

$$hconf(X) = \frac{s(i_1, i_2, \dots, i_k)}{\max[s(i_1), s(i_2), \dots, s(i_k)]}$$

where $s(X)$ is the support of an itemset X . Those itemsets X that satisfy $hconf(X) \geq \alpha$, where α is a user-defined threshold, are known as hyperclique patterns. These patterns have been shown to be useful for various applications, including clustering [60], semi-supervised classification [59], data cleaning [58], and finding functionally coherent sets of proteins [57].

2.3 Error-Tolerant Patterns

Traditional association patterns are obtained using a strict definition of support that requires every item in a frequent itemset to appear in each supporting transaction. In real-life datasets, this limits the recovery of frequent itemsets as they are fragmented due to random noise and other errors in the data. Motivated by such considerations, various methods [62, 38, 47, 27, 11] have been proposed recently to discover approximate frequent itemsets, which are also often called error-tolerant itemsets (ETIs). These methods tolerate some error by allowing itemsets in which a specified fraction of the items can be missing. This error tolerance can either be specified on the complete submatrix of the collection of items and transactions or in each row and/or column. For instance, in Figure 1(c), the itemset shown is a error tolerant itemset with 20% error tolerance in each row. It is important to note that each of the proposed definitions of error tolerant patterns will lead to a traditional frequent itemset if their error-tolerance is set to 0. For a detailed comparison of several algorithms proposed to discover ETIs from binary data sets, and their extensions, the reader is referred to our previous work [19].

2.4 Discriminative Pattern Mining

A variety of real-life data sets include information about which transactions belong to which of some pre-specified classes, i.e., class label information. For

such data sets, patterns of considerable interest are those that occur with disproportionate support or frequency in some classes versus the others. These patterns have been investigated under various names such as emerging patterns [16], contrast sets [4] and discriminative patterns [9,18,10], but we will refer to them as discriminative patterns. Consider the example in Figure 2, which displays a sample dataset, in which there are 14 items and two classes, each containing 10 instances (transactions). In this data set, four discriminative patterns can be observed: $P_1 = \{i_1, i_2, i_3\}$, $P_2 = \{i_5, i_6, i_7\}$, $P_3 = \{i_9, i_{10}\}$ and $P_4 = \{i_{12}, i_{13}, i_{14}\}$. Intuitively, P_1 and P_4 are interesting patterns that occur with differing frequencies in the two classes, while P_2 and P_3 are uninteresting patterns that have a relatively uniform occurrence across classes. Furthermore, we observe that P_4 is a discriminative pattern whose individual items are also highly discriminative, while P_1 is a discriminative pattern whose individual items are not.

Discriminative patterns have been shown to be useful for improving the classification performance for transaction data sets when combinations of features have better discriminative power than individual features [9,55,53]. Discriminative pattern mining has the potential to discover groups of genes or SNPs that are individually not informative but are highly associated with a phenotype when considered as a group.

		P1			P2				P3			P4			
		i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}
Class A	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1
	0		0	0	1	1	1	0	0	0	0	0	1	1	1
	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1
	1	1	1	0	1	1	1	0	0	0	0	0	1	1	1
	1	1	1	0	1	1	1	0	1	1	0	0	1	1	1
	1	1	1	0	1	1	1	0	1	1	0	0	1	1	1
	1	1	1	0	1	1	1	0	1	1	0	0	1	1	1
	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1
Class B	1	1	0	0	0	1	0	0	0	0	0	0	1	1	1
	1	0	0	0	1	1	1	0	1	1	0	0	0	0	0
	1	0	0	0	1	1	1	0	1	1	0	1	0	1	0
	1	0	1	0	1	1	1	0	1	1	0	0	0	1	0
	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0
	0	1	1	0	1	1	1	0	0	0	0	0	0	0	0
	0	1	1	0	1	1	1	0	0	0	0	0	0	0	1
	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 2. Example of interesting discriminative patterns (P_1 , P_4) and uninteresting patterns (P_2 , P_3)

3 Case Study: Association Analysis-Based Pre-processing of Protein Interaction Networks

One of the most promising forms of biological data that are used to study the functions and other properties of proteins at a genomic scale are protein interaction networks. These networks provide a global view of the interactions between various proteins that are essential for the accomplishment of most protein functions. Due to the importance of the knowledge of these interactions, several high-throughput methods have been proposed for discovering them [25]. In fact, several standardized databases, such as DIP [56] and GRID [7] have now been

set up to provide systematic access to protein interaction data collected from a wide variety of experiments and sources.

It is easy to see that a protein interaction network can be represented as an undirected graph, where proteins are represented by nodes and protein-protein interactions as edges. Due to this systematic representation, several computational approaches have been proposed for the prediction of protein function from these graphs [36,45,46,39,54,26,31]. These approaches can be broadly categorized into four types, namely neighborhoodbased, global optimization-based, clustering-based and association analysis-based. Due to the rich functional information in these networks, several of these approaches have produced very good results, particularly those that use the entire interaction graph simultaneously and use global optimization techniques to make predictions [31,54]. Indeed, recently, some studies have started using protein interaction networks as benchmarks for evaluating the functional relationships between two proteins, such as [63].

However, despite the advantages of protein interaction networks, they have several weaknesses which affect the quality of the results obtained from their analysis. The most prominent of these problems is that of noise in the data, which manifests itself primarily in the form of spurious or false positive edges [44,21]. Studies have shown that the presence of noise has significant adverse affects on the performance of protein function prediction algorithms [15]. Another important problem facing the use of these networks is their incompleteness, i.e., the absence of biologically valid interactions even from large sets of interactions [54,21]. This absence of interactions from the network prevents even the global optimization-based approaches from making effective use of the network beyond what is available, thus leading to a loss of potentially valid predictions.

A possible approach to address these problems is to transform the original interaction graph into a new weighted graph such that the weights assigned to the edges in the new graph more accurately indicate the reliability and strength of the corresponding interactions, and their utility for predicting protein function. The usefulness of hypercliques in noise removal from binary data [58], coupled with the representation of protein interaction graphs as a binary adjacency matrix to which association analysis techniques can be applied, motivated Pandey et al. [37] to address the graph transformation problem using an approach based on h -confidence measure discussed earlier. This measure is used to estimate the common neighborhood similarity of two proteins P_1 and P_2 as

$$h\text{-confidence}(P_1, P_2) = \min \left(\frac{|N_{P_1} \cap N_{P_2}|}{|N_{P_1}|}, \frac{|N_{P_1} \cap N_{P_2}|}{|N_{P_2}|} \right) \quad (1)$$

where N_{P_1} and N_{P_2} denote the sets of neighbors of P_1 and P_2 respectively. As discussed earlier, this definition of h -confidence is only applicable to binary data or, in the context of protein interaction graphs, unweighted graphs. However, the notion of h -confidence can be readily generalized to networks where the edges carry real-valued weights indicating their reliability. In this case, Equation 1 can be conveniently modified to calculate h -confidence(P_1, P_2) by making the following substitutions: (1) $|N_{P_1}| \rightarrow$ sum of weights of edges incident on P_1 (similarly for P_2) and (2) $|N_{P_1} \cap N_{P_2}| \rightarrow$ sum of minimum of weights of each pair

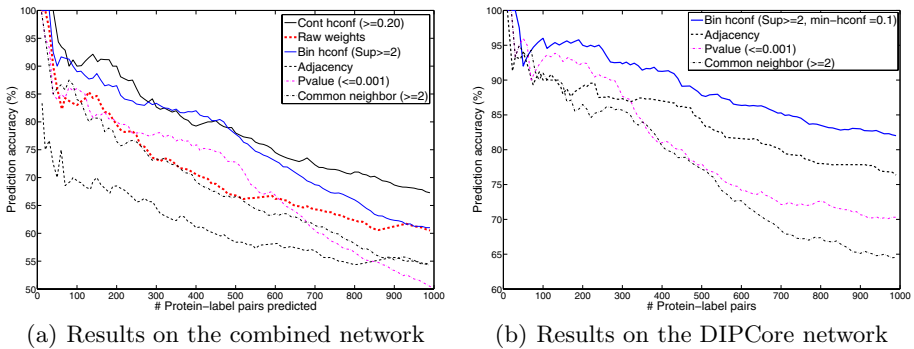


Fig. 3. Comparison of performance of various transformed networks and the input networks (Best viewed in color and a larger size)

of edges that are incident on a protein P from both P_1 and P_2 . In both these cases, the h – confidence measure is guaranteed to fall in the range $[0, 1]$.

Now, with this definition, it is hypothesized that protein pairs having a high h – confidence score are expected to have a valid interaction between them, since a high value of the score indicates a high common neighborhood similarity, which in turn reflects greater confidence in the network structure for that interaction. For the same reason, interactions between protein pairs having a low h – confidence score are expected to be noisy or spurious. Accordingly, Pandey *et al* [37] proposed the following graph transformation approach for pre-processing available interaction data sets. First, using the input interaction network $G = (V, E)$, the h – confidence measure is computed between each pair of constituent proteins, whether connected or unconnected by an edge in the input network. Next, a threshold is applied to drop the protein pairs with a low h – confidence to remove spurious interactions and control the density of the network. The resultant graph $G' = (V, E')$ is hypothesized to be the less noisy and more complete version of G , since it is expected to contain fewer noisy edges, some biologically viable edges that were not present in the original graph, and more accurate weights on the remaining edges.

In order to evaluate the efficacy of the resultant networks for protein function prediction, the original and the transformed graphs were provided as input to the FunctionalFlow algorithm [31], which is a popular graph theory-based algorithm for predicting protein function from interaction networks. The performance was also compared with transformed versions generated using other common neighborhood similarity measures for such networks, such as Samanta *et al* [45]’s p-value measure. Figure 3 shows the performance of this algorithm on these transformed versions of two standard interaction networks, namely the *combined* data set constructed by combining several popular yeast interaction data sets (combined) and weighted using the EPR Index tool [14], and the other being a confident subset of the DIP database [14] (DIPCore). The performance is evaluated using the accuracy of the top scoring 1000 predictions of the functions of the constituent proteins generated by a five-fold cross-validation procedure,

where the functional annotations are obtained from the classes at depth two of the FunCat functional hierarchy [43].

The results in Figure 3 show that for both the data sets, the h -confidence-based transformed version(s) substantially outperform the original network and the other measures for this task. The margin of improvement on the highly reliable DIPCore data set is almost consistently 5% or above, which is quite significant. Similar results are observed using the complete precision-recall curves. The interested reader is referred to [37] for more details on the methodology used and the complete results.

4 Concluding Remarks and Directions for Future Research

Association analysis has proved to be a powerful approach for analyzing traditional market basket data, and has even been found useful for some problems in bioinformatics in a few instances. However, there are a number of other important problems in bioinformatics, such as finding biomarkers using dense data like SNP data and real-valued data like gene-expression data, where such techniques could prove to be very useful, but cannot currently be easily and effectively applied.

An important example of patterns that are not effectively captured by the traditional association analysis framework and its current extensions, is a group of genes that are co-expressed together across a subset of conditions in a gene expression data set. Such patterns have often been referred to as *biclusters*. Figure 4 illustrates a classification of biclusters proposed by Madeira et al. [28]. They classified different types of biclusters into four categories: (i) biclusters with constant values (Figure 4(a)), (ii) biclusters with constant rows or columns (Example of a bicluster with constant rows is shown in Figure 4(b)), (iii) biclusters with coherent values, i.e., each row and column is obtained by addition or multiplication of the previous row and column by a constant value (Figure 4(c)), and (iv) biclusters with coherent evolutions, where the direction of change of values is important rather than the coherence of the values (Figure 4(d)). Each of these types of biclusters hold different types of significance for discovering important knowledge from gene expression data sets.

Since gene expression data is real-valued, traditional association techniques can not be directly applied since they are designed for binary data. Methods

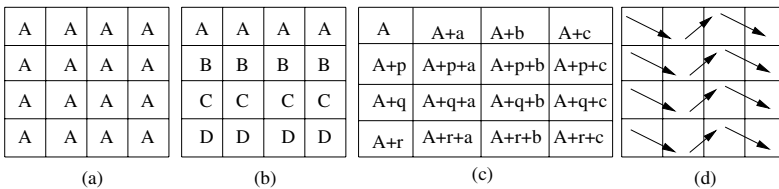


Fig. 4. Types of Biclusters: (a) Biclusters with constant values (b) Biclusters with constant rows (c) Biclusters with coherent values (additive model) (d) Biclusters with coherent evolutions

for transforming these data sets into binary form (for example, via discretization [5,13,30]) often suffer from loss of critical information about the actual. Hence, a variety of other techniques have been developed for and/or applied to this problem. These approaches include a wide variety of clustering techniques: ordinary partitional and hierarchical clustering, subspace clustering, biclustering/co-clustering, projective clustering, and correlation clustering. In addition, a variety of biclustering algorithms have been developed for finding such patterns from gene expression data, such as ISA [6], Cheng and Church's algorithm [12] and SAMBA [51], and more recently, for genetic interaction data [41]. Although these algorithms are often able to find useful patterns, they suffer from a number of limitations. The most important limitation is an inability to efficiently explore the entire search space for such patterns without resorting to heuristic approaches that compromise the completeness of the search. Pandey et al. [34] have presented one of the first methods for directly mining association patterns from real-valued data, particularly gene expression data, that does not suffer from the loss of information often faced by discretization and other data transformation approaches [34]. These techniques are able to discover all patterns satisfying the given constraints, unlike the biclustering algorithms that may only be able to discover a subset of these patterns. There are several open opportunities for designing better algorithms for addressing this problem.

Another challenge that has inhibited the use of association analysis in bioinformatics—even when the data is binary—is the density of several types of data sets. Algorithms for finding association patterns often break down when the data becomes dense because of the large number of patterns generated, unless a high support threshold is used. However, with a high threshold, many interesting, low-support patterns are missed. One particularly important category of applications with dense data are applications involving class labels, such as finding connections between genetic variations and disease. Consider the problem of finding connections between genetic variations and disease using binarized version of SNP-genotype data, which is 33% dense by design, since each subject must have one of the three variations of SNP pairs: *major-major*, *major-minor*, *minor-minor*. Traditional algorithms that do not utilize the class label information for pruning can only find patterns at high support, thus missing the low support patterns that are typically of great interest in this domain. In fact, most of the existing techniques for this problem only apply univariate analysis and rank individual SNPs using measures like p-value, odds ratio etc [3,22]. There are some approaches like Multi-Dimensionality Reduction (MDR) [42] and Combinatorial Partitioning Methods (CPM) [32], which are specially designed to identify groups of SNPs. However, due to their brute-force way of searching the exponential search space, these approaches also can only be applied to data sets with small number of SNPs (typically of the order of few dozen SNPs). Also, existing discriminative pattern mining algorithms [4,9,18,10] are only able to prune infrequent non-discriminative patterns, not the frequent non-discriminative patterns, which is the biggest challenge for dense data sets like SNP data and gene expression data. New approaches should be designed to

enable discriminative pattern mining on dense and high dimensional data, where effectively making use of class label information for pruning is crucial. Extension of association analysis based approaches to effectively use the available class label information for finding low-support discriminative patterns is a promising direction for future research.

In conclusion, significant scope exists for future research on designing novel association analysis techniques for complex biological data sets and their associated problems. Such techniques will significantly aid in realizing the potential of association analysis for discovering novel knowledge from these data sets and solve important bioinformatics problems.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Proc. SIGMOD, pp. 207–216 (1993)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. VLDB, pp. 487–499 (1994)
3. Balding, D.: A tutorial on statistical methods for population association studies. *Nature Reviews Genetics* 7(10), 781 (2006)
4. Bay, S., Pazzani, M.: Detecting group differences: Mining contrast sets. *DMKD* 5(3), 213–246 (2001)
5. Becquet, C., et al.: Strong-association-rule mining for large-scale gene-expression data analysis: a case study on human sage data. *Genome Biology* 3 (2002)
6. Bergmann, S., Ihmels, J., Barkai, N.: Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical Review* 67 (2003)
7. Breitkreutz, B.-J., Stark, C., Tyers, M.: The GRID: the General Repository for Interaction Datasets. *Genome Biology* 4(3), R23 (2003)
8. Ceglar, A., Roddick, J.F.: Association mining. *ACM Comput. Surv.* 38(2), 5 (2006)
9. Cheng, H., Yan, X., Han, J., Hsu, C.-W.: Discriminative frequent pattern analysis for effective classification. In: Proc. IEEE ICDE, pp. 716–725 (2007)
10. Cheng, H., Yan, X., Han, J., Yu, P.: Direct mining of discriminative and essential graphical and itemset features via model-based search tree. In: Proc. ACM SIGKDD International Conference, pp. 230–238 (2008)
11. Cheng, H., Yu, P.S., Han, J.: Ac-close: Efficiently mining approximate closed itemsets by core pattern recovery. In: Proceedings of the 2006 IEEE International Conference on Data Mining, pp. 839–844 (2006)
12. Cheng, Y., Church, G.: Biclustering of Expression Data. In: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology table of contents, pp. 93–103. AAAI Press, Menlo Park (2000)
13. Creighton, C., Hanash, S.: Mining gene expression databases for association rules. *Bioinformatics* 19(1), 79–86 (2003)
14. Deane, C.M., Salwinski, L., Xenarios, I., Eisenberg, D.: Protein interactions: two methods for assessment of the reliability of high throughput observations. *Mol Cell Proteomics* 1(5), 349–356 (2002)
15. Deng, M., Sun, F., Chen, T.: Assessment of the reliability of protein–protein interactions and protein function prediction. In: Pac. Symp. Biocomputing, pp. 140–151 (2003)

16. Dong, G., Li, J.: Efficient mining of emerging patterns: Discovering trends and differences. In: Proceedings of the 2001 ACM SIGKDD International Conference, pp. 43–52 (1999)
17. Eisenberg, D., Marcotte, E.M., Xenarios, I., Yeates, T.O.: Protein function in the post-genomic era. *Nature* 405(6788), 823–826 (2000)
18. Fan, W., Zhang, K., Cheng, H., Gao, J., Yan, X., Han, J., Yu, P.S., Verscheure, O.: Direct discriminative pattern mining for effective classification. In: Proc. IEEE ICDE, pp. 169–178 (2008)
19. Gupta, R., Fang, G., Field, B., Steinbach, M., Kumar, V.: Quantitative evaluation of approximate frequent pattern mining algorithms. In: Proceeding of the 14th ACM SIGKDD Conference, pp. 301–309 (2008)
20. Han, J., Pei, J., Yin, Y., Mao, R.: Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery* 8(1), 53–87 (2004)
21. Hart, G.T., Ramani, A.K., Marcotte, E.M.: How complete are current yeast and human protein-interaction networks? *Genome. Biol.* 7(11), 120 (2006)
22. Hirschhorn, J.: Genetic Approaches to Studying Common Diseases and Complex Traits. *Pediatric Research* 57(5 Part 2), 74R (2005)
23. Klemettinen, M., Mannila, H., Toivonen, H.: Rule Discovery in Telecommunication Alarm Data. *J. Network and Systems Management* 7(4), 395–423 (1999)
24. Kuramochi, M., Karypis, G.: An efficient algorithm for discovering frequent sub-graphs. *IEEE Trans. on Knowl. and Data Eng.* 16(9), 1038–1051 (2004)
25. Legrain, P., Wojcik, J., Gauthier, J.-M.: Protein–protein interaction maps: a lead towards cellular functions. *Trends Genet.* 17(6), 346–352 (2001)
26. Lin, C., Jiang, D., Zhang, A.: Prediction of protein function using common-neighbors in protein-protein interaction networks. In: Proc. IEEE Symposium on BionInformatics and BioEngineering (BIBE), pp. 251–260 (2006)
27. Liu, J., Paulsen, S., Sun, X., Wang, W., Nobel, A., Prins, J.: Mining Approximate Frequent Itemsets In the Presence of Noise: Algorithm and Analysis. In: Proc. SIAM International Conference on Data Mining (2006)
28. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Trans. Comput. Biol. Bioinf.* 1(1), 24–45 (2004)
29. Martinez, R., Pasquier, N., Pasquier, C.: GenMiner: mining non-redundant association rules from integrated gene expression data and annotations. *Bioinformatics* 24(22), 2643–2644 (2008)
30. McIntosh, T., Chawla, S.: High confidence rule mining for microarray analysis. *IEEE/ACM Trans. Comput. Biol. Bioinf.* 4(4), 611–623 (2007)
31. Nabieva, E., Jim, K., Agarwal, A., Chazelle, B., Singh, M.: Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics* 21(suppl. 1), i1–i9 (2005)
32. Nelson, M., Kardia, S., Ferrell, R., Sing, C.: A Combinatorial Partitioning Method to Identify Multilocus Genotypic Partitions That Predict Quantitative Trait Variation. *Genome Research* 11(3), 458–470 (2001)
33. Nguyen, D.V., Arpat, A.B., Wang, N., Carroll, R.J.: DNA microarray experiments: biological and technological aspects. *Biometrics* 58(4), 701–717 (2002)
34. Pandey, G., Atluri, G., Steinbach, M., Kumar, V.: Association analysis for real-valued data: Definitions and application to microarray data. Technical Report 08-007, Department of Computer Science and Engineering, University of Minnesota (March 2008)

35. Pandey, G., Atluri, G., Steinbach, M., Kumar, V.: Association analysis techniques for discovering functional modules from microarray data. *Nature Proceedings*, Presented at ISMB, SIG Meeting on Automated Function Prediction (2008), <http://dx.doi.org/10.1038/npre.2008.2184.1>
36. Pandey, G., Kumar, V., Steinbach, M.: Computational approaches for protein function prediction: A survey. Technical Report 06-028, Department of Computer Science and Engineering, University of Minnesota (October 2006)
37. Pandey, G., Steinbach, M., Gupta, R., Garg, T., Kumar, V.: Association analysis-based transformations for protein interaction networks: a function prediction case study. In: *Proceedings of the 13th ACM SIGKDD International Conference*, pp. 540–549 (2007)
38. Pei, J., Tung, A., Han, J.: Fault-tolerant frequent pattern mining: Problems and challenges. In: *Workshop on Research Issues in Data Mining and Knowledge Discovery* (2001)
39. Pereira-Leal, J.B., Enright, A.J., Ouzounis, C.A.: Detection of functional modules from protein interaction networks. *Proteins* 54(1), 49–57 (2003)
40. Pfaltz, J., Taylor, C.: Closed set mining of biological data. In: *Workshop on Data Mining in Bioinformatics (BIOKDD)* (2002)
41. Pu, S., Ronen, K., Vlasblom, J., Greenblatt, J., Wodak, S.J.: Local coherence in genetic interaction patterns reveals prevalent functional versatility. *Bioinformatics* 24(20), 2376–2383 (2008)
42. Ritchie, M., et al.: Multifactor dimensionality reduction reveals high-order interactions among estrogen- metabolism genes in sporadic breast cancer. *Am. J. Hum. Genet.* 69(1), 1245–1250 (2001)
43. Ruepp, A., et al.: The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Res.* 32(18), 5539–5545 (2004)
44. Salwinski, L., Eisenberg, D.: Computational methods of analysis of protein-protein interactions. *Curr. Opin. Struct. Biology* 13(3), 377–382 (2003)
45. Samanta, M.P., Liang, S.: Predicting protein functions from redundancies in large-scale protein interaction networks. *Proc. Natl. Acad. Sci. U.S.A.* 100(22), 12579–12583 (2003)
46. Schwikowski, B., Uetz, P., Fields, S.: A network of protein-protein interactions in yeast. *Nature Biotechnology* 18(12), 1257–1261 (2000)
47. Seppanen, J., Mannila, H.: Dense itemsets. In: *KDD*, pp. 683–688 (2004)
48. Seshasayee, A.S.N., Babu, M.M.: Contextual inference of protein function. In: Subramaniam, S. (ed.) *Encyclopaedia of Genetics and Genomics and Proteomics and Bioinformatics*. John Wiley and Sons, Chichester (2005)
49. Tan, P., Kumar, V., Srivastava, J.: Selecting the right interestingness measure for association patterns. In: *Proceedings of the eighth ACM SIGKDD International Conference*, pp. 32–41 (2002)
50. Tan, P.-N., Steinbach, M., Kumar, V.: *Introduction to Data Mining*. Addison-Wesley, Reading (2005)
51. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. *Bioinformatics* 18(suppl. 1), 136–144 (2002)
52. Tramontano, A.: *The Ten Most Wanted Solutions in Protein Bioinformatics*. CRC Press, Boca Raton (2005)
53. van Vliet, M., Klijn, C., Wessels, L., Reinders, M.: Module-based outcome prediction using breast cancer compendia. *PLoS ONE* 2(10), 1047 (2007)

54. Vazquez, A., Flammini, A., Maritan, A., Vespignani, A.: Global protein function prediction from protein-protein interaction networks. *Nat. Biotechnology* 21(6), 697–700 (2003)
55. Wang, J., Karypis, G.: Harmony: Efficiently mining the best rules for classification. In: *Proceedings of SIAM International Conference on Data Mining*, pp. 205–216 (2005)
56. Xenarios, I., Salwinski, L., Duan, X.J., Higney, P., Kim, S.-M., Eisenberg, D.: DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Research* 30(1), 303–305 (2002)
57. Xiong, H., He, X., Ding, C., Zhang, Y., Kumar, V., Holbrook, S.R.: Identification of functional modules in protein complexes via hyperclique pattern discovery. In: *Proc. Pacific Symposium on Biocomputing (PSB)*, pp. 221–232 (2005)
58. Xiong, H., Pandey, G., Steinbach, M., Kumar, V.: Enhancing data analysis with noise removal. *IEEE Trans. on Knowl. and Data Eng.* 18(3), 304–319 (2006)
59. Xiong, H., Steinbach, M., Kumar, V.: Privacy leakage in multi-relational databases via pattern based semi-supervised learning. In: *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 355–356. ACM, New York (2005)
60. Xiong, H., Steinbach, M., Tan, P., Kumar, V.: HICAP: Hierarchical Clustering with Pattern Preservation. In: *Proceedings of the 4th SIAM International Conference on Data Mining*, pp. 279–290 (2004)
61. Xiong, H., Tan, P.-N., Kumar, V.: Hyperclique pattern discovery. *Data Min. Knowl. Discov.* 13(2), 219–242 (2006)
62. Yang, C., Fayyad, U., Bradley, P.: Efficient discovery of error-tolerant frequent itemsets in high dimensions. In: *Proc. ACM SIGKDD*, pp. 194–203 (2001)
63. Yona, G., Dirks, W., Rahman, S., Lin, D.M.: Effective similarity measures for expression profiles. *Bioinformatics* 22(13), 1616–1622 (2006)
64. Zaki, M., Ogihara, M.: Theoretical foundations of association rules. In: *3rd ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery* (June 1998)

Analyzing and Interrogating Biological Networks (Abstract)

Eric Banks, Elena Nabieva, Bernard Chazelle, Ryan Peterson,
and Mona Singh

Department of Computer Science and Lewis-Sigler Institute for Integrative Genomics,
Princeton University, Princeton, NJ 08544

High-throughput experimental and computational approaches to characterize proteins and their interactions have resulted in large-scale biological networks for many organisms, from bacteria to yeast to human. These complex networks are comprised of a number of distinct types of interactions: these include interactions between proteins that interact physically, that participate in a synthetic lethal or epistatic relationship, that are coexpressed, or where one phosphorylates or regulates another. Though incomplete and noisy, these networks provide a holistic view of the functioning of the cell and, with appropriate computational analysis and experimental work, have significant potential for helping to uncover cellular principles as well as protein functions and pathways.

We introduce a framework for explicitly incorporating known attributes of individual proteins into the analysis of biological networks. We conceptualize this with *network schemas*. A network schema describes a group of proteins with specific characteristics along with the desired topology and types of interactions among them. A schema's matches (or instances) in an interactome are subgraphs of the interaction network that are made up of proteins having the specified characteristics which interact with one another as dictated by the schema's topology. For example, a schema associated with signaling might be a linear path of kinases interacting in succession; its instances in the baker's yeast *S. cerevisiae* include portions of the pheromone response and filamentous growth pathways.

In the first part of this work, we show how to utilize network schemas to uncover organizational units corresponding to recurring means with which diverse biological processes are carried out. We develop algorithms for systematically uncovering recurring, over-represented schemas in physical interaction networks. We apply our methods to the *S. cerevisiae* interactome, focusing on schemas consisting of proteins described via sequence motifs and molecular function annotations and interacting with one another in one of four basic network topologies. We identify hundreds of recurring and over-represented network schemas of various complexity, and demonstrate via graph-theoretic representations how more complex schemas are organized in terms of their lower-order constituents. The uncovered schemas span a wide-range of cellular activities, with many signaling and transport related higher-order schemas. We establish the functional importance of the schemas by showing that they correspond to functionally cohesive sets of proteins, are enriched in the frequency with which they have instances in the human interactome, and are useful for predicting protein function.

In the second part of this work, we introduce a system, NetGrep, for searching protein interaction networks for matches to general network schema queries. The NetGrep system allows querying with schemas where proteins are described via a diverse set of features, including Prosite family, Pfam motif, SMART domain, Supfam superfamily and Gene Ontology annotations. Proteins may also be specified via particular protein IDs, homology to other proteins, regular expressions over amino acids, or with unions or intersections over any of the previously described features. By utilizing these protein attributes in combination with physical, genetic, phosphorylation, regulatory, and/or coexpression interactions (as available for the organism of interest via high-throughput experiments), the network schema queries allowed in NetGrep generalize many previously studied interaction patterns, including domain-domain interactions, signaling and regulatory pathways, and network motifs. Network schemas can also be naturally extended to handle approximate matches by specifying optional nodes. Although the search problem arising from finding all the matches for a given network schema is a case of the computationally difficult subgraph isomorphism problem, we have been able to develop algorithms that take advantage of schema characteristics for biological networks. As a result, NetGrep's core algorithms are extremely fast in practice for queries with up to several thousand matches in the interactomes studied. Our algorithms can thus enable new analysis that characterizes networks with respect to the types and numbers of interaction patterns found.

From Architecture to Function (and Back) in Bio-networks

Vladimir Filkov

UC Davis, Department of Computer Science, Davis CA 95616, USA
filkov@cs.ucdavis.edu

Abstract. Living organisms exhibit systemic or emergent behavior due in large part to the combinatorial interdependencies among the components in the complex molecular systems that comprise them. Biological networks encode these interdependencies into convenient structural representations, enabling their analysis and understanding. Examples include: synergistic behavior of transcription factors during gene regulation, where several of them act together atomically to elicit gene expression response; the modular architecture of biological networks and its relation to function; and the enrichment of certain small topological features in the networks, e.g. triangles, called network motifs. Studying the connection between biological network architecture and function is made possible by novel methods enabling correct statistical and combinatorial accounting of topological features in the network systems.

My focus in the past few years has been on characterizing biological networks in terms of the underlying local topologies or building blocks, and linking those building blocks to observable function in living organisms. Here I describe a set of tools and techniques that help us do that, together with results and insights from our studies. The techniques described come from various scientific disciplines that have had to deal with networked systems for some time, like social sciences, statistical mechanics, and complex network theory among others. In all case we have adapted these tools to biological network analysis and either extended or improved upon them.

1 Determining the Topological Building Blocks of Biological Networks

To characterize biological networks in terms of a small set of topological building blocks we use a family of statistical models called exponential random graph (ERG) models, also known as p^* models [1]. ERG models provide a tool to study the way that a network's global structure (and function) depends on a wide range of local structures such as number of edges, node degree, number of triangles, etc. In this approach, the full network is modeled as a random sample from a distribution which represents the observed data best, in which the local features are explanatory variables combined linearly, and the dependence of the dependent network on this combination is exponential. The coefficients in the model can be interpreted as the prominence or importance of a local feature (variable) for the whole network. Advanced data fitting methods based on Maximum Likelihood Estimation are used to fit the coefficients in the model to the

data. We have been able to successfully apply ERG models to identify key features in biological networks and to subsequently provide a novel way of classifying biological networks based on the prevalence of their local features. The flexibility, in terms of the number of available local feature choices, and scalability, in terms of the network sizes, make this approach ideal for statistical modeling of biological networks.

2 Associating Topologies with Organism Properties

Our next goal was to map or associate discovered prominent topologies to organisms properties, or even specific function or phenotypes in organisms. To that end, in our recent work [2] we have undertaken an automated approach using various topological metrics from network theory to characterize a collection of various kinds of biological networks and show how the most informative metrics can be used to associate them with organism characteristics. We built a comprehensive assembly of networks of different types for many different organisms which we characterized using a suite of network theory metrics, so as to comprehensively compare them simultaneously, allowing for a much more in-depth evaluation of network models than is possible using the commonly existing practice of comparing one or two particular properties (most commonly the degree distribution). Using data mining techniques (PCA and clustering) we show that multiple parameters are necessary and often sufficient to characterize the variability in networks meaningfully. Hierarchical linear regression models on logarithmically adjusted network metrics data successfully identified subsets of network metrics which associate surprisingly well with organism characteristics, viz. organism class, network type, genome size, GC content, and modularity. Such classification or cataloging of biologically associated topological features can yield vocabularies which can be consulted for a given subnetwork, or function.

3 Using the Building Blocks to Predict Missing Links

Knowing that certain network types are characterized with specific topologies, e.g. large number of shared neighbors between highly connected nodes, we investigated if we can leverage that knowledge to improve the prediction of missing links in incomplete protein-protein interaction networks and gene regulation networks. In fact, when ranked according to their participation in certain topological patterns, and predicting the top ranked links, the prediction accuracy for missing links grew ten fold compared to considering gene expression data alone.

The above is joint work with my students Z. Saul, S. Roy, and V. Missirian.

References

1. Saul, Z.M., Filkov, V.: Exploring Biological Network Structure Using Exponential Random Graph Models. *Bioinformatics* 23, 2604 (2007)
2. Roy, S., Filkov, V.: The strong associations between organism characteristics and network architecture. arXiv:0901.0775

A New Machine Learning Approach for Protein Phosphorylation Site Prediction in Plants

Jianjiong Gao^{1,2}, Ganesh Kumar Agrawal^{2,3}, Jay J. Thelen^{2,3},
Zoran Obradovic⁴, A. Keith Dunker⁵, and Dong Xu^{1,2,*}

¹ Department of Computer Science

² C.S. Bond Life Sciences Center

³ Department of Biochemistry, University of Missouri, Columbia, Missouri 65211

⁴ Center for Information Science and Technology, Temple University,
Philadelphia, PA 19122

⁵ Center for Computational Biology and Bioinformatics, Indiana University Schools of
Medicine and Informatics, Indianapolis, IN 46202

Tel.: +1 573-884-1887; Fax: +1 573-882-8318

xudong@missouri.edu

Abstract. Protein phosphorylation is a crucial regulatory mechanism in various organisms. With recent improvements in mass spectrometry, phosphorylation site data are rapidly accumulating. Despite this wealth of data, computational prediction of phosphorylation sites remains a challenging task. This is particularly true in plants, due to the limited information on substrate specificities of protein kinases in plants and the fact that current phosphorylation prediction tools are trained with kinase-specific phosphorylation data from non-plant organisms. In this paper, we proposed a new machine learning approach for phosphorylation site prediction. We incorporate protein sequence information and protein disordered regions, and integrate machine learning techniques of k-nearest neighbor and support vector machine for predicting phosphorylation sites. Test results on the PhosPhAt dataset of phosphoserines in *Arabidopsis* and the TAIR7 non-redundant protein database show good performance of our proposed phosphorylation site prediction method.

Keywords: Protein Phosphorylation, Phosphoproteomics, *Arabidopsis*, Protein Disorder, KNN, SVM.

1 Introduction

Reversible protein phosphorylation is one of the most pervasive posttranslational modification mechanisms, regulating diverse cellular processes in various organisms. It has been estimated that about 30% of all proteins in a cell are phosphorylated at any given time [1]. In recent years, publicly available protein phosphorylation data have rapidly accumulated due to large-scale, mass spectrometry studies of protein phosphorylation in different organisms [2-6] and the development of associated phosphorylation web resources [7-11].

* Corresponding author.

Protein phosphorylation can occur on serine, threonine and tyrosine residues, as well as histidine and aspartate residues in the case of two-component phosphorelays. However, O-linked phosphorylation, specifically on serine residues, is the most common form of phosphorylation in eukaryotes. Despite the increasing number of large-scale phosphorylation studies, experimental identification of phosphorylation sites is still a difficult and time-consuming task. Therefore, more efficient methods for predicting phosphorylation sites *in silico* are desirable. A number of phosphorylation site prediction tools have been developed, including Scansite 2.0 [12], NetPhosK [13], PredPhospho [14], DISPHOS [15], KinasePhos [16], PPSP [17], pkaPS [18], Predikin [19], GPS 2.0 [20], AutoMotif [21] and CRPhos [22]. However, these tools have limitations when predicting phosphorylation sites in plants for two major reasons: (1) they were trained mostly on phosphorylation data from non-plant—mainly mammalian organisms; (2) all of them except DISPHOS, were trained on kinase-specific phosphorylation data and aimed to predict kinase substrate specificities. Meanwhile, the phosphorylation data in plants are not as well annotated as those in mammals, with much less information available on the specificity of phosphorylation sites and their corresponding kinases. Therefore, there is a clear need to train a reliable phosphorylation predictor in plants given the increased frequency of protein kinases in plant genomes and the lack of knowledge about their substrate specificities. With the recently released PhosPhAt database, potential phosphoserines were predicted for the Arabidopsis protein database TAIR7 [23] by support vector machine (SVM) trained on the experimental data collected in the database [10]. Nevertheless, there is room for improvement in prediction accuracy.

In this paper, we proposed a new machine learning approach for phosphorylation site prediction in plants, which integrates features from protein disorder information, nearest neighbors of known phosphorylation sites, and amino acid frequencies in the surrounding sequences of phosphorylation sites to train an SVM for phosphorylation site prediction. The key differences between our method and the previous study [10] are that we incorporated protein disorder prediction and nearest neighbor information in the prediction. A previous study demonstrated that disorder information significantly improved the discrimination between phosphorylation and non-phosphorylation sites [15]. With increasing volume of empirical phosphorylation sites, it is advantageous to use nearest neighbor information. Test results on the PhosPhAt [10] dataset of phosphoserines and the TAIR7 [23] non-redundant protein database indeed shows the remarkable performance of our proposed phosphorylation prediction method.

2 Materials and Methods

Phosphorylation site prediction can be formulated as a binary classification problem, namely each serine/threonine/tyrosine can be classified as either phosphorylation site or non-phosphorylation site. As with all general binary classification problems, there are three key issues: (1) a well-collected and curated dataset including positive and negative data; (2) a set of effective features to characterize the common patterns in

each category and the differences between the two categories; (3) a classifier trained from the known data, capable of making reliable predictions for new data. In this study, datasets were extracted from the TAIR7 protein database and PhosPhAt phosphorylation database as discussed in Section 2.1. Outputs from a protein disorder predictor, outputs from k-nearest neighbor predictions and amino acid frequencies around the phosphorylation sites were taken as features as discussed in Section 2.2. We used SVM as the classifier.

2.1 Phosphorylation Dataset

Phosphorylation data in the model organism *Arabidopsis thaliana* collected in PhosPhAt [10] and the *Arabidopsis thaliana* protein database TAIR7 were utilized in this study. Sequences with high similarities were first removed from TAIR7 to build a non-redundant (NR) protein database using BLASTClust in the BLAST package version 2.2.19 with a sequence identity threshold of 30%. As a result, 12,018 representative proteins remain in the TAIR NR database. The PhosPhAt phosphorylation data were then incorporated resulting in 1152 phosphoproteins in the TAIR NR database, which contain 2050 phosphorylation sites, including 1818 phosphoserines, 130 phosphothreonines and 102 phosphotyrosines. We only study phosphoserine events in this paper because of the large number of available data for training and testing. However, the proposed method can be applied to all types of phosphorylation sites.

A 25-residue-long amino acid sequence surrounding each phosphoserine with the phosphoserine in the middle was extracted from each phosphoprotein in the TAIR NR database. Phosphoserines with upstream or downstream less than 12 residues were discarded. As a result, we retrieved a positive set with 1671 sequences surrounding phosphoserines. Similarly, the 433,744 sequences surrounding the non-phosphoserines (serines other than the phosphoserines) were assumed to be the negative set. Although not all these sites are necessarily true negatives, it is reasonable to believe that the vast majority of them are.

2.2 Feature Extraction and Selection

2.2.1 K-Nearest Neighbor Features

Both of the positive and negative sets are very diverse at the sequence level. However, clusters may exist in the positive set, since each phosphorylation site is the substrate of a specific protein kinase, and one kinase could target multiple substrates. It is well known that substrates of the same kinase may share similar patterns in sequence [24]. To take advantage of the cluster information when predicting phosphorylation for a new site (represented by its surrounding sequence), we extracted features from its similar sequences in both positive and negative sets retrieved by a k-nearest neighbor (KNN) algorithm as the following procedure.

- i) For a new sequence s , find its k nearest neighbors (NN) in positive and negative sets respectively according to the sequence distance measure defined as follows. For two protein sequences $s_1 = \{s_1(-w), s_1(-w+1), \dots, s_1(w-1), s_1(w)\}$ and $s_2 = \{s_2(-w), s_2(-w+1), \dots, s_2(w-1), s_2(w)\}$, define the distance $Dist(s_1, s_2)$ between s_1 and s_2 as

$$Dist(s_1, s_2) = 1 - \frac{\sum_{i=-w}^w Sim(s_1(i), s_2(i))}{2w+1} \quad (1)$$

where w is the length of left/right window ($w=12$) and Sim —amino acid similarity matrix—is derived from the normalized BLOSUM62 [25]:

$$Sim(a, b) = \frac{Blosum(a, b) - \min\{Blosum\}}{\max\{Blosum\} - \min\{Blosum\}} \quad (2)$$

where a and b are two amino acids, $Blosum$ is the BLOSUM62 matrix, and $\max/\min\{Blosum\}$ represent the largest/smallest number in the $Blosum$ matrix.

- ii) The corresponding KNN feature is then extracted as follows
 - a) Calculate the average distances from the new sequence s to the k nearest neighbors in the positive and negative sets, respectively.
 - b) Calculate KNN score—the ratio of the average distance to the nearest neighbors in the positive set against that in the negative set.
- iii) To take advantage of different properties of neighbors with different similarities, repeat (i) and (ii) for different k 's to get multiple features for the phosphorylation predictor. In this paper, k was chosen to be 0.1%, 0.2%, 0.5%, 1%, 2%, 5% and 10% of the size of positive/negative sets, and thus 7 KNN scores were extracted as features for the phosphorylation prediction.

2.2.2 Protein Disorder Features

It was observed that sites of posttranslational modifications, including protein phosphorylation sites, are frequently located within disordered regions [15, 26]. In [15], the disorder prediction results for the phosphorylation sites were employed as features to construct a phosphorylation predictor—DISPHOS. In this study, we extracted the disorder information for all surrounding residues of each phosphorylation site and combined them to form a set of disorder features in SVM. The procedure is as follows:

- i) For each protein in the TAIR NR database, predict its disordered region using VSL2B [27].
- ii) Extract the disorder prediction scores for the surrounding residues in both positive and negative sets, and thus form a vector of 25 scores.
- iii) Take the average scores surrounding the sites with different window sizes as features for the phosphorylation predictor. In this paper, we chose the window sizes to be 1, 9 and 25, and thus three disorder features were extracted for each sequence.

2.2.3 Amino Acid Frequency Features

In [15], Iakoucheva *et al.* analyzed the amino acid composition of the surrounding sequences of phosphorylation sites and found that rigid, buried, neutral amino acids (W, C, F, I, Y, V and L) are significantly depleted, while flexible, surface-exposed amino acids (S, P, E, K) are significantly enriched. This conclusion was confirmed by this study as illustrated in Section 3.3. This fact makes the amino acid frequencies

good candidates as features for phosphorylation site prediction. In this paper, all 20 amino acid frequencies in each 25-residue sequence were extracted as features for the phosphorylation predictor.

3 Results and Discussions

3.1 KNN Scores as Features

The KNN scores were extracted as features according to the procedure described in Section 2.2.1. A KNN score for a sequence of interest actually compares its average distance (or dissimilarity) to the nearest neighbors (NNs) in the positive set with that in the negative set. A score smaller than 1 means the sequence is more similar to the positive set; a score larger than 1 means more similar to the negative set. The smaller the KNN score, the more similar the sequence is to known phosphorylation sites, and thus the more likely it contains a phosphorylation site.

Figure 1 compares the KNN scores of phosphoserines with non-phosphoserines. Overall the phosphoserines have smaller KNN scores than non-phosphoserines. All of the phosphoserines' average KNN scores with different sizes of NNs are smaller than 1, which means overall the sequences in the positive set are more similar to their NNs in the positive set as expected. It is worth mentioning that such similarities are not due to protein homology as there is no significant sequence similarity between any two proteins in our non-redundant dataset. This finding confirms that phosphorylation-related clusters may exist in the positive set as discussed in Section 2.2.1.

Interestingly, all of the non-phosphoserines' average KNN scores are around 1, which means overall the sequences in the negative set are not predominantly more similar to NNs in either the positive or negative sets. This is not surprising, since phosphorylation-related clusters are unlikely to exist in the negative set, and thus the sequences in the negative set have similar chance to find close neighbors in either positive or negative set.

In short, KNN scores capture the cluster information in phosphoserines, and hence distinguish them from non-phosphoserines. Therefore, KNN scores are suitable to serve as features for the phosphorylation site prediction. The prediction performance of KNNs scores will be demonstrated in Section 3.4.

3.2 Protein Phosphorylation and Disorder

In this section, we will demonstrate that phosphoserines in the dataset we used are predominantly overrepresented in disordered regions, and hence confirm the effectiveness of the disorder scores as features for phosphorylation prediction. Figures 2(A) and 2(B) plot the histograms of the disorder scores of phosphoserines and non-phosphoserines' surrounding residues, respectively. From Fig. 2(A), the number of phosphoserines increases exponentially when the disorder score increases from 0 to 1; the number of phosphoserines with disorder scores larger than 0.9 is much higher than those in the other sub-ranges. In contrast, from Fig. 2(B), there is no such a pattern for the non-phosphoserines. The number of non-phosphoserines with disorder

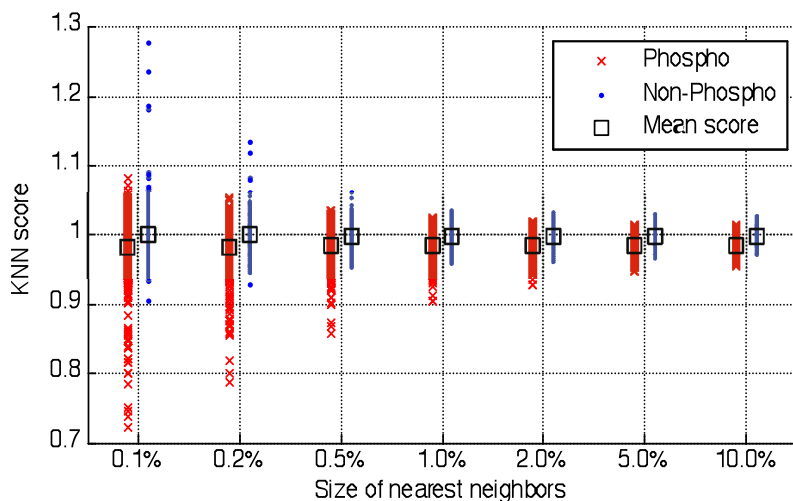


Fig. 1. Comparison of KNN scores in the positive set (1671 sequences around phosphoserines) and those in the negative set (randomly selected 1671 sequences around non-phosphoserines). The horizontal axis represents the size of nearest neighbors (in percentage relative to the size of positive/negative set). The vertical axis represents the KNN score. Each KNN score for one sequence is represented by 'x' (positive data) or dot (negative data). Each square symbol stands for the mean value of KNN scores for each size of NNs.

scores larger than 0.9 is slightly higher than those in the other sub-ranges. This may be because some phosphoserines were not discovered by the experiments in [10] and as a result were incorrectly classified as non-phosphoserines. Alternatively, this could also reflect the general preference of serine in disordered regions. In any case, it is clear that phosphoserines in this dataset are significantly overrepresented in disordered regions. In fact, the majority (~89%) of the phosphoserines have a disorder score larger than 0.5 (Note that VSL2B predicts a residue in the disordered region when its predicted value is larger than 0.5), while this percentage is only ~57% for non-phosphoserines.

3.3 Amino Acid Frequency Features

In this section, we will study the amino acid composition surrounding the phosphoserines. In Figure 3, from left to right, the amino acids vary from being depleted to being enriched in the surrounding sequences of phosphoserines. Similarly as observed in [15], amino acids C, W, Y, F, H, I, L are depleted around phosphoserines, while D, E, R, P and K are enriched. However, S is not significantly enriched around the phosphoserines in this dataset, in contrast to the previous study [15]. The different composition of amino acids surrounding phosphoserines and non-phosphoserines justifies the use of amino acid frequencies as features for the phosphorylation predictor.

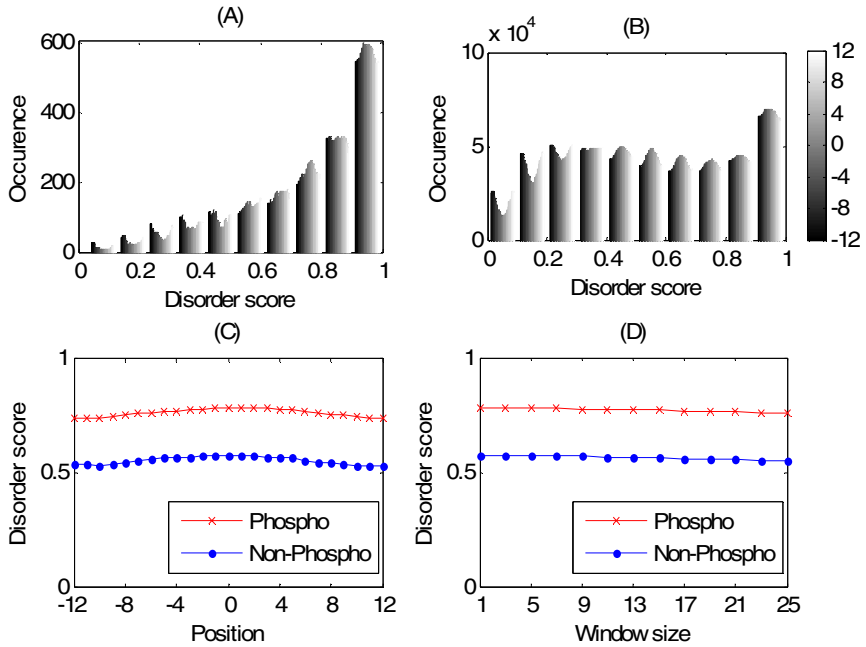


Fig. 2. Preference of phosphorylation sites (serines) in disordered regions. **(A)** Histogram of disorder scores of residues around phosphoserines (1671 in total). The horizontal axis represents the disorder score predicted by VSL2B, divided into 10 sub-ranges from 0 to 1; the vertical axis represents the occurrence (the number of sequences) in the corresponding disorder sub-range. Different grayscale from dark to white in each bar stand for 25 different positions in the window from the upstream -12 to downstream 12 as indicated in the *color bar* on the top right. **(B)** Histogram of disorder scores of residues around non-phosphoserines (433,744 in total). **(C)** Disorder scores in the positive and negative sets. The horizontal axis represents the 25 positions (-12 to 12); the vertical axis represents the mean disorder score in the positive set ('x') or the negative set (dot). **(D)** Average disorder scores over windows of different lengths. The horizontal axis represents the window size over which to take average of the disorder scores for each surrounding sequence. The vertical axis represents the mean of those average scores.

3.4 SVM Training and Testing

In this study, an SVM was trained as the classifier between phosphoserines and non-phosphoserines. The SVM^{light} Version 6.02 [28] was used. The parameters were optimized as '-t 2 -g 1 -c 10 -x 1', which means selecting the kernel as radial basis function with gamma equal to 1, setting C—the tradeoff between training error and margin to 10, and computing the leave-one-out estimate.

As mentioned in Section 2.1, there are 1671 serines in the positive set and 433,744 in the negative set. Testing of the proposed method was performed using the following procedure:

- i) Randomly select 1671 samples from the negative set, together with the positive set, and form a balanced dataset of 3342 samples.

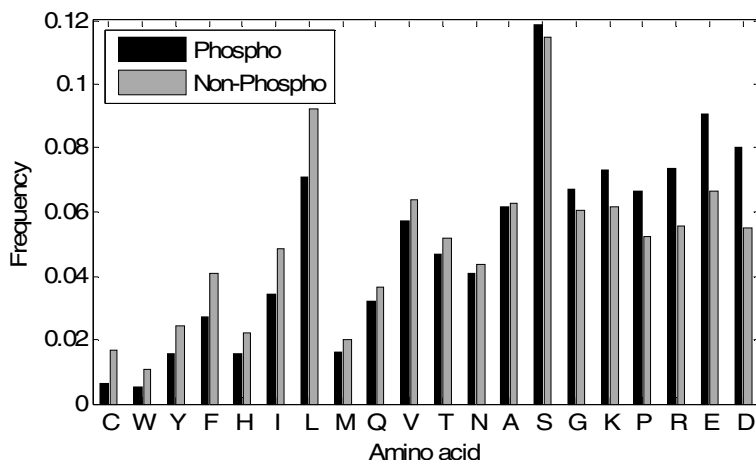


Fig. 3. Amino acid frequencies in the positive and negative sets (the serines in the middle of the 25 residues were excluded; all positive and negative data were used). The vertical axis represents the amino acid frequency. The horizontal axis represents the 20 amino acids sorted in ascending order by the ratio between the amino acid's frequency in the positive set (black) and that in the negative set (gray).

- ii) Perform a 10-fold cross validation test: the dataset was partitioned into 10 subsets; a single subset was retained as validation data and the other 9 sets as training data; the cross-validation process is then repeated 10 times, with each subset used exactly once as the validation data. The 10 results were then combined to produce an average estimation.
- iii) Note: in each training/test, the disorder and frequency features remained the same. However, the KNN features of each training or validation needed to be re-extracted from the training data, and every time the training data was changed.

The above testing procedure was performed on each separate set of features (amino acid features only, disorder features only, or KNN features only) and combined features (all three sets of features together) 10 times each. Table 1 shows the area under receiver operating characteristic (ROC) curve (AUC) for each test of each set of features, and also the mean AUCs and the standard deviations. Figure 4 shows the mean ROC curves for these tests.

Table 1 and Figure 4 show that all of the three sets of features provide certain predictive powers, but the combined features gave the best test results with the smallest variance (standard deviation) among the 10 random tests. This indicates that combining various features yields more accurate and robust prediction. When testing the features separately, the disorder features were not performed as accurately as the KNN features and frequency features. This may be partially due to fact that all the data came from the same species (*Arabidopsis*). It is unclear whether similar performance can be maintained for cross-species prediction (e.g., training with *Arabidopsis*

Table 1. Prediction performance (AUC) for 10 random tests for different sets of features

Test	Frequency only	Disorder only	KNN only	Combined
1	0.754	0.722	0.816	0.840
2	0.769	0.707	0.806	0.825
3	0.768	0.729	0.812	0.825
4	0.758	0.723	0.796	0.830
5	0.769	0.727	0.813	0.830
6	0.764	0.730	0.794	0.823
7	0.765	0.733	0.813	0.829
8	0.734	0.719	0.819	0.827
9	0.771	0.715	0.816	0.828
10	0.759	0.715	0.793	0.817
Mean	0.761	0.722	0.808	0.827
Standard Deviation	0.011	0.008	0.010	0.006

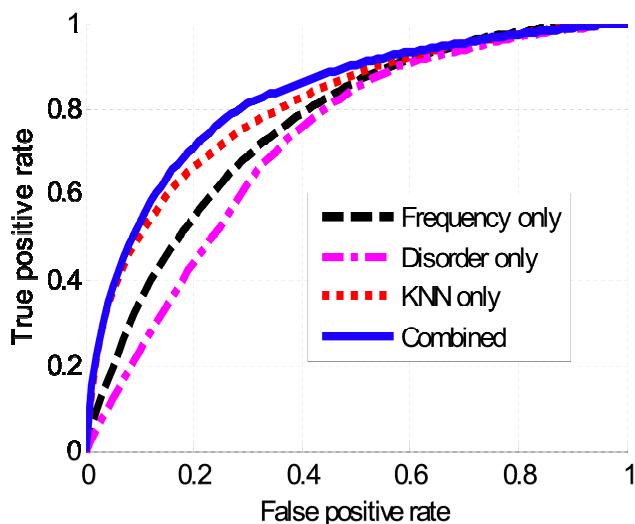


Fig. 4. Mean receiver operating characteristic curves of 10 random tests for different sets of features. The *horizontal axis* represents the false positive rate (the fraction of misclassified samples in the randomly selected negative set); the *vertical axis* represents the true positive rate (the fraction of correctly detected samples in the positive set).

data and predicting phosphorylation sites in soybean). There, the disordered information may be more generic and species-independent.

The phosphoserine predictor in [10] gave a performance of AUC around 0.81 on the redundant *Arabidopsis* TAIR7 protein dataset. It is worth mentioning that for the redundant dataset, the test results of our method achieved 0.84-0.85 on AUC, as KNN may find sequence neighbors in close homologs of the query protein.

4 Conclusion and Future Work

In this paper, we developed a new approach for predicting protein phosphorylation sites in plants. We treated phosphorylation site prediction as a binary classification problem, and employed machine learning techniques to solve it. Multiple features were first extracted from the dataset, including features from nearest neighbors, protein disordered regions and amino acid frequencies. We demonstrated that phosphoserines in the PhosPhAt dataset are predominantly overrepresented in disordered regions. An SVM was then trained based on these features, and used to predict phosphorylation sites in new data. Our method combined both KNN to take advantage of similar known sequence fragments around phosphorylation sites to query protein sequences and SVM to account for other generic features. Test results show good performance of this proposed phosphorylation prediction method. As more phosphorylation sites are experimentally identified, the accuracy of our method is expected to increase automatically.

In future work, we plan to apply our method on phosphothreonines and phosphotyrosines, as well as to the whole proteomes of *Arabidopsis* and other plant species. We will also develop a standalone application and a web service based on this work.

Acknowledgments. This work was supported by the funding from the National Science Foundation-Plant Genome Research Program [grant number DBI-0604439 awarded to JJT] and the National Institute of Health [grant number R21/R33 GM078601 awarded to DX]. The authors wish to thank Dr. Predrag Radivojac, Dr. Jingfen Zhang, and Zhiqian He for helpful discussion and technical assistance.

References

1. Steen, H., Jeganathirajah, J.A., Rush, J., Morrice, N., Kirschner, M.W.: Phosphorylation analysis by mass spectrometry: myths, facts, and the consequences for qualitative and quantitative measurements. *Mol. Cell Proteomics* 5(1), 172–181 (2006)
2. Olsen, J.V., Blagoev, B., Gnäd, F., Macek, B., Kumar, C., Mortensen, P., Mann, M.: Global, in vivo, and site-specific phosphorylation dynamics in signaling networks. *Cell* 127, 635–648 (2006)
3. Villén, J., Beausoleil, S.A., Gerber, S.A., Gygi, S.P.: Large-scale phosphorylation analysis of mouse liver. *Proc. Natl. Acad. Sci. USA* 104, 1488–1493 (2007)
4. Chi, A., Huttenhower, C., Geer, L.Y., Coon, J.J., Syka, J.E., Bai, D.L., Shabanowitz, J., Burke, D.J., Troyanskaya, O.G., Hunt, D.F.: Analysis of phosphorylation sites on proteins from *Saccharomyces cerevisiae* by electron transfer dissociation (ETD) mass spectrometry. *Proc. Natl. Acad. Sci. USA* 104, 2193–2198 (2007)
5. Benschop, J.J., Mohammed, S., O’Flaherty, M., Heck, A.J., Slijper, M., Menke, F.L.: Quantitative Phosphoproteomics of Early Elicitor Signaling in *Arabidopsis*. *Mol Cell Proteomics* 6, 1198–1214 (2007)
6. Sugiyama, N., Nakagami, H., Mochida, K., Daudi, A., Tomita, M., Shirasu, K., Ishihama, Y.: Large-scale phosphorylation mapping reveals the extent of tyrosine phosphorylation in *Arabidopsis*. *Mol. Syst. Biol.* 4, 193 (2008)

7. Diella, F., Gould, C.M., Chica, C., Via, A., Gibson, T.J.: Phospho.ELM: a database of phosphorylation sites—update 2008. *Nucleic Acids Res.* 36(Database issue), D240–D244 (2008)
8. Gnad, F., Ren, S., Cox, J., Olsen, J.V., Macek, B., Orosi, M., Mann, M.: PHOSIDA (phosphorylation site database): management, structural and evolutionary investigation, and prediction of phosphosites. *Genome Biol.* 8, R250 (2007)
9. Tchieu, J.H., Fana, F., Fink, J.L., Harper, J., Nair, T.M., Niedner, R.H., Smith, D.W., Steube, K., Tam, T.M., Veretnik, S., Wang, D., Gribskov, M.: The PlantsP and PlantsT Functional Genomics Databases. *Nucleic Acids Res.* 31, 342–344 (2003)
10. Heazlewood, J.L., Durek, P., Hummel, J., Selbig, J., Weckwerth, W., Walther, D., Schulze, W.X.: PhosPhAT: a database of phosphorylation sites in *Arabidopsis thaliana* and a plant-specific phosphorylation site predictor. *Nucleic Acids Res.* 36(Database issue), D1015–D1021 (2008)
11. Gao, J., Agrawal, G.K., Thelen, J.J., Xu, D.: P3DB: a plant protein phosphorylation database. *Nucleic Acids Res.* 37(Database issue), D960–D962 (2009)
12. Obenaus, J.C., Cantley, L.C., Yaffe, M.B.: Scansite 2.0: Proteome-wide prediction of cell signaling interactions using short sequence motifs. *Nucleic Acids Res.* 31(13), 3635–3641 (2003)
13. Blom, N., Sicheritz-Ponten, T., Gupta, R., Gammeltoft, S., Brunak, S.: Proteomics. Prediction of post-translational glycosylation and phosphorylation of proteins from the amino acid sequence 4(6), 1633–1649 (2004)
14. Kim, J.H., Lee, J., Oh, B., Kimm, K., Koh, I.: Prediction of phosphorylation sites using SVMs. *Bioinformatics* 20(17), 3179–3184 (2004)
15. Iakoucheva, L.M., Radivojac, P., Brown, C.J., O'Connor, T.R., Sikes, J.G., Obradovic, Z., Dunker, A.K.: The importance of intrinsic disorder for protein phosphorylation. *Nucleic Acids Res.* 32(3), 1037–1049 (2004)
16. Huang, H.D., Lee, T.Y., Tzeng, S.W., Horng, J.T.: KinasePhos: a web tool for identifying protein kinase-specific phosphorylation sites. *Nucleic Acids Res.* 33(Web Server issue), W226–W229 (2005)
17. Xue, Y., Li, A., Wang, L., Feng, H., Yao, X.: PPSP: prediction of PK-specific phosphorylation site with Bayesian decision theory. *BMC Bioinformatics* 7, 163 (2006)
18. Neuberger, G., Schneider, G., Eisenhaber, F.: pkaPS: prediction of protein kinase A phosphorylation sites with the simplified kinase substrate binding model. *Biol. Direct.* 2, 1 (2007)
19. Saunders, N.F., Kobe, B.: The Predikin webserver: improved prediction of protein kinase peptide specificity using structural information. *Nucleic Acids Res.* 36(Web Server issue), W286–W290 (2008)
20. Xue, Y., Ren, J., Gao, X., Jin, C., Wen, L., Yao, X.: GPS 2.0, a tool to predict kinase-specific phosphorylation sites in hierarchy. *Mol. Cell Proteomics* 7(9), 1598–1608 (2008)
21. Plewczynski, D., Tkacz, A., Wyrwicz, L.S., Rychlewski, L., Ginalski, K.: AutoMotif Server for prediction of phosphorylation sites in proteins using support vector machine: 2007 update. *J. Mol. Model* 14(1), 69–76 (2008)
22. Dang, T.H., Van Leemput, K., Verschoren, A., Laukens, K.: Prediction of kinase-specific phosphorylation sites using conditional random fields. *Bioinformatics* 24(24), 2857–2864 (2008)
23. Swarbreck, D., Wilks, C., Lamesch, P., Berardini, T.Z., Garcia-Hernandez, M., Foerster, H., Li, D., Meyer, T., Muller, R., Ploetz, L., Radenbaugh, A., Singh, S., Swing, V., Tissier, C., Zhang, P., Huala, E.: The *Arabidopsis* Information Resource (TAIR): gene structure and function annotation. *Nucleic Acids Res.* 36(Database issue), D1009–D1014 (2008)

24. Kennelly, P.J., Krebs, E.G.: Consensus sequences as substrate specificity determinants for protein kinases and protein phosphatases. *J. Biol. Chem.* 266, 15555–15558 (1991)
25. Henikoff, S.: Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA* 89, 10915–10919 (1992)
26. Dunker, A.K., Oldfield, C.J., Meng, J., Romero, P., Yang, J.Y., Chen, J.W., Vacic, V., Obradovic, Z., Uversky, V.N.: The unfoldomics decade: an update on intrinsically disordered proteins. *BMC Genomics* 9(Suppl. 2), S1 (2008)
27. Obradovic, Z., Peng, K., Vucetic, S., Radivojac, P., Dunker, A.K.: Exploiting heterogeneous sequence properties improves prediction of protein disorder. *Proteins* 61(suppl 7), 176–182 (2005)
28. Joachims, T.: SVM^{light} Version 6.0.2 (2008), <http://svmlight.joachims.org>

Assembly of Large Genomes from Paired Short Reads

Benjamin G. Jackson¹, Patrick S. Schnable², and Srinivas Aluru^{1,2}

¹ Department of Electrical and Computer Engineering

² Center For Plant Genomics

Iowa State University, Ames, IA 50011, USA

Abstract. The *de novo* assembly of genomes from high-throughput short reads is an active area of research. Several promising methods have been recently developed, with applicability largely restricted to the smaller and less complex bacterial genomes. In this paper, we present a method for assembling large genomes from high-coverage paired short reads. Our method exploits large distributed memory and parallelism available on multiprocessor systems to handle memory-intensive phases of the algorithm, effectively allowing scaling to large genomes. We present parallel algorithms to construct a bidirected string graph that is several orders of magnitude smaller than the raw sequence data and to extract features from paired reads. We also present a heuristic method that uses these features to guide the extension of partial graph traversals corresponding to large genomic contigs. In addition, we propose a simple model for error correction and derive a lower bound on the coverage needed for its use. We present a validation of our framework with short reads from *D. melanogaster* and *S. cerevisiae* synthetically generated at 300-fold coverage. Assembly of the *D. melanogaster* genome resulted in large contigs (50% of the genome covered by contigs larger than 102Kb), accurate to 99.9% of the bases, in under 4 hours of wall clock time on a 512-node Blue Gene/L.

1 Introduction

For nearly three decades from its invention, Sanger sequencing - which produces 700 to 1000 base-pair reads - dominated the field of DNA sequencing and genome assembly. New developments in high-throughput short read sequencing are proving a disruptive technology that allows concurrent generation of millions of reads at a significantly lower per base cost, albeit with limitations on read length (25-50 bp typically, with the exception of 454, which can produce 150 bp reads). Several such platforms are available and seeing rapid adoption in the community (454 Life Sciences system [13], Illumina Solexa [1], Applied Biosystems SOLiD [17], and Helicos Biosciences Heliscope [22]).

Short-read technologies were initially aimed at resequencing individuals when a template genome of the species is known. This allows aligning the reads to the reference genome, avoiding a *de novo* assembly, and provides an easy way for

biological analysis such as identifying Single Nucleotide Polymorphisms (SNPs). The Solexa system has been used for resequencing [2] [19], identifying repeats [21], and characterizing population diversity [16].

There has been considerable recent interest in the more technically challenging problem of *de novo* genome sequencing. Given the multimillion dollar expense associated with Sanger read based genome sequencing projects, high-throughput technologies offer the only hope of sequencing a much larger number of species. Also, *de novo* assembly is important in cases where significant genomic rearrangements are expected, such as when sequencing multiple inbred lines of the same plant species. In response to these needs, several short read assemblers have recently been developed – ALLPATHS [3], Euler-SR [4], SHARCGS [5], Shorty [8], an assembler by Medvedev *et al.* [14], SSAKE [20], and Velvet [23].

Traditionally, the overlap-layout-consensus paradigm has been the mainstay of Sanger read based sequencing projects, whereby long overlaps between reads provide much of the information to shape the assembly. Though graph-based methods that permit a more global view when resolving repeats have been developed (de Bruijn [9] [18] and string graph models [15]), their reach has not extended beyond bacterial genomes due to significant memory requirements.

With short reads eliminating the reliability of read overlaps in predicting genomic co-location, a revival of graph-based methods has underpinned the development of short-read assemblers. However, the memory limitations seen previously are further exacerbated due to the high coverage needed to compensate for shorter read lengths. As a result, short-read *de novo* assembly has been demonstrated on relatively small genome sizes, ranging from single BACs [5] [20] [23] to bacterial genomes with a few million bases [4] [7] [14]. Perhaps the largest reported assembly (39 million bases) was produced in 2 days using a workstation with 64 GB memory [3].

In this paper, we present a short-read sequence assembly framework that can successfully assemble large genomes with high coverage. To do so, we rely on parallel algorithms and the large distributed memory afforded by high performance parallel computers for the memory-intensive phases of the assembly. We present parallel methods for constructing a bidirected de Bruijn graph of k -molecules and converting this graph into a bidirected string graph. We present a parallel method for computing features that summarize the distances between paired reads. We also create weak traversal constraints derived from k -mer frequency along each edge. We present an algorithm that finds paths corresponding to contigs by starting with unambiguous long edges as seeds, and then extending each path using heuristic rules that rely on the computed features. As the string graph is many orders of magnitude smaller than the initial sequence data, and there are only a few thousand features per edge, this final phase of assembly can be carried out sequentially for many genomes. Our method advances the state of the art in short read assembly from the confines of *prokaryotic* genomes. For example, we demonstrate a 99.9% accurate assembly of the *Drosophila Melanogaster* genome in four hours, with 50% of the genome covered by contigs of length at least 102Kb.

2 Linking Coverage to Error Correction

The two most crucial technical challenges in accurate short read assembly are eliminating sequencing errors and accurately reconstructing genomic repeats. High coverage, randomness in error location, and a low error rate are helpful when dealing with errors. Paired reads obtained by sequencing both ends of fragments of known approximate size provide distance constraints that are instrumental in identifying correct walks in the graph and resolving repeats.

As with other methods, the method we propose is sensitive to errors unless they are identified prior to assembly. In a manner similar to that proposed by Pevzner *et al.* [18], we deal with errors by analyzing k -mer frequency. We wish to find a threshold such that, with high probability, all k -mers with frequency below this threshold are artifacts due to sequencing errors. Correspondingly, all real k -mers should occur at a rate above this threshold. We make the simplifying assumption that the sampling of the genome and the sequencing error are independent, uniform, random processes. For the initial analysis, we ignore the increased frequencies of certain k -mers due to the presence of repeats, but address this subsequently.

Let g the length of the genome, l be the length of each read, r be the substitution error rate per base, and c be the coverage rate per base. We describe a k -mer as a tuple $s = \langle p_1, p_2, \dots, p_k \rangle$. An *edit profile* is the corresponding tuple $\langle c_1, c_2, \dots, c_k \rangle$ with $c_i \in [0, 3]$. The probability $P(c_i = 0)$ is $1 - r$ (i.e., base called correctly), while the probabilities $P(c_i = 1) = P(c_i = 2) = P(c_i = 3) = \frac{r}{3}$ (corresponding to each of the three incorrect base call possibilities).

We are interested in two classes of edit profiles: the identity profile, which has probability $(1 - r)^k$; and the profiles corresponding to a single edit, each with probability $\frac{r}{3}(1 - r)^{(k-1)}$. We ignore other profiles given their low probabilities for practical values of r . The expected rate at which the identity profile occurs at some location in the genome is $\lambda_p = \binom{c(l-k)}{l} (1 - r)^k$. The expected rate at which a single error edit profile occurs is $\lambda_d = \binom{c(l-k)}{l} \left(\frac{r}{3}\right) (1 - r)^{k-1}$.

The number of times a particular k -mer (the identity profile at a particular position) is seen in the data is a Poisson process, with the expected number of k -mers seen exactly t times given by the equation:

$$\mathcal{C}_t = \left(\frac{\lambda_p^t e^{-\lambda_p}}{t!} \right) g$$

The expected number of times a single base k -mer edit is seen exactly t times is defined similarly:

$$\mathcal{E}_t = \left(\frac{\lambda_d^t e^{-\lambda_d}}{t!} \right) 3kg$$

Given a genome of length g and an error rate r , we wish to find coverage c such that good k -mers can be separated from bad k -mers by some threshold τ with high probability. We create a 3-dimensional plot of \mathcal{C}_t and \mathcal{E}_t given c and

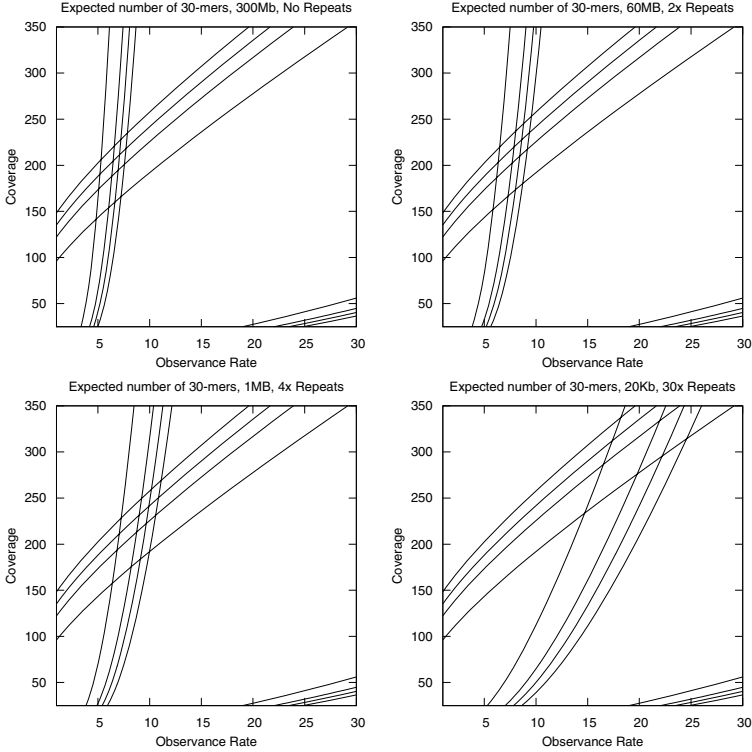


Fig. 1. Contour lines for C_t and \mathcal{E}_t when plotted against c and t . We plot $\log_{10} C_t = \{-2, -1, 0, \text{ and } 2\}$ for genome length 300Mb, read length of 40bp, 1% error, and $k=30$. We also plot $\log_{10} \mathcal{E}_t = \{-2, -1, 0, \text{ and } 2\}$ for a hypothetical genome repeat decomposition, superimposed against C_t . We show in the upper left a plot of \mathcal{E}_t for 300Mb of unique sequence. We show in the upper right a plot for 60Mb of sequence repeated twice. We show in the lower left a plot for 1Mb of sequence repeated 4 times. Finally, we show in the lower right a plot for 20Kb repeated 30 times. These plots indicate that with 1% sequencing error rate, 30-mers can likely be differentiated using a simple threshold method at 250-fold to 300-fold coverage.

τ , as shown in the upper left quadrant of Figure [1](#). Observe that if the genome is unique, a good separation of 30-mers can be achieved with 200-fold coverage of length 40bp reads with 1% error.

We can update this analysis given the presence of sampling bias and repeats by observing that both can be modeled as non-uniform coverage of some genome with only unique k -mers. We can analyze the k -mers by separating this genome into sets of k -mers with similar coverage. Our task is to find a single threshold that separates real k -mers from errors in all sets simultaneously. As shown in Figure [1](#), for a genome of length 300Mb, a 1% error rate, and an average read length of 40, we can expect that 300-fold coverage will achieve separation of 30-mers for many repeats.

3 Parallel Bidirected String Graph Construction

The first step in the assembly method is the parallel construction of a bidirected string graph such that genomic contigs that could be inferred from the reads correspond to paths in the graph. We begin by constructing a bidirected de Bruijn graph with nodes corresponding to k -molecules and edges corresponding to $(k+1)$ -molecules. We assume a distributed memory parallel model where each processor has its own local memory and access to other processors' memory is mediated by an interconnection network. We have previously described a method for constructing a bidirected string graph in parallel [10, 11], which we summarize here.

In a bidirected graph, each edge has two directions, one for each endpoint. There are four possible ways of connecting edges for an ordered pair of nodes (u, v) : $u \triangleright \triangleright v$, $u \triangleleft \triangleleft v$, $u \triangleright \triangleleft v$, and $u \triangleleft \triangleright v$. A valid graph traversal requires that endpoint directions be satisfied when passing through a node; if we enter a node on an in arrow, we must exit on an out arrow, and vice versa.

Consider each k -molecule in the data (by considering both strands associated with each k -mer). We call the lexicographically larger of the two strands the positive (or representative) strand, and the other strand the negative strand. Consider a de Bruijn graph, where each k -molecule corresponds to a node in the graph, and two nodes are connected if they share a common $(k-1)$ -molecule. There are four ways of connecting edges in the de Bruijn graph, and we use a bidirected edge to capture these options, as shown in Figure 2. If the positive strand of the underlying molecule moves into the edge, the corresponding arrowhead points away from the node. Similarly, if the negative strand of the underlying molecule moves into the edge, the corresponding arrowhead points into the node.

We construct the de Bruijn graph edge-by-edge, by looking at all $(k+1)$ -molecules in the input data. The representative strand of each molecule is stored as a $4(k+1)$ bit integer in a distributed array. To operate within the confines of available memory, the sequences are read in stages, updating this representative list at the end of each phase with the frequency of each molecule in the data, using parallel sort [6] as the supporting operation.

Also to make efficient use of memory, data is processed in two phases. In the first phase, we record the frequency of all observed $(k+1)$ -molecules. After enough data has been considered, all real $(k+1)$ -molecules will have been



Fig. 2. The four ways in which k -molecule nodes can be connected in the bidirected de Bruijn graph of the data, with the corresponding edge labels that will be used in creating the string graph

observed at least once with high probability. At this point, we continue with the second phase, in which only the frequencies of previously seen $(k + 1)$ -molecules are updated. After all data has been considered, molecules with frequency below the threshold indicated in the previous section are discarded.

Once all true $(k + 1)$ -molecules are found, the graph can be determined directly by constructing an edge from each representative. This graph is nearly a de Bruijn graph, but there can exist nodes in this graph that are not connected but would be in a true de Bruijn graph, due to the existence of maximal $k - 1$ length repeats in the underlying genome [10]. We will refer to the graph constructed in this manner as a de Bruijn graph despite this subtle difference.

We can label the edges in this graph with two characters, each corresponding to the first base from each strand of the $(k + 1)$ -molecule. We are interested in compacting chains in the de Bruijn graph, as we have described extensively in [11]. We solve this problem by transforming the chain compaction problem to the parallel undirected list ranking problem. During edge compaction, we also concatenate the edge labels, to produce assembled contigs.

The result of this process is a bidirected string graph [15]. The bidirected string graph we create is the lowest order graph homeomorphic with the de Bruijn graph. Each edge in this graph is labeled by some genomic sequence, which may or may not be repeated multiple times in the genome. We arbitrarily assign each edge in the graph a forward direction which will be used during the processing of paired reads. Some traversal of this graph corresponds to the entire assembled genome, although there are many possible traversals.

4 Paired Read Processing

The reads in the sequence data come in pairs, each read pair coming from the two ends of a sheared DNA fragment. Typically fragment lengths fall into a specified range. Our method allows for the consideration of multiple such fragment ranges, which we call fragment types, and assumes that reads are classified accordingly.

In the bidirected string graph $G = \{E, V\}$, edges e_i and e_j correspond to genomic sequences s_i and s_j , each sequence occurring one or more times in the genome. If the genomic distance between these two sequences falls within some fragment range, there will be evidence of these two sequences' relative location in the sequence data. We will summarize this information as a set of features we term *partial $(k + 1)$ -pair clusters*, and use these features to finish assembly.

Definition 1. A **position** in the bidirected graph G is a tuple of the form $p = \langle e, f \rangle$, with $e \in E$, $f \in \mathbb{N}$, $0 \leq f < \|e\|$. The field f corresponds to a position along the edge in its forward direction, indexed from zero.

Note 1. By construction of the string graph, there is a bijection between valid $(k + 1)$ -molecules in the input and the set of all positions in the graph. Therefore we use $p(m)$ to denote the position corresponding to $(k + 1)$ -molecule m , and $p(m).e$ and $p(m).f$ to denote the corresponding fields.

Definition 2. A **read pair** is a tuple of the form $\langle R_1, R_2, z \rangle$, where R_1 and R_2 are the reads and z is the fragment type.

Definition 3. A $(k+1)$ -**pair** is a tuple of the form $\pi = \langle m_1, m_2, z \rangle$, where m_1 and m_2 are molecules.

Note 2. We use $\lceil z \rceil$ to denote the largest possible distance between observed $(k+1)$ -molecules when reading the ends of a fragment of type z , and $\lfloor z \rfloor$ to denote the smallest possible distance. If z_{min} is the minimum length of fragment type z , z_{max} the maximum fragment length, and l the maximum read length, $\lfloor z \rfloor = z_{min} - 2l + (k+1)$, and $\lceil z \rceil = z_{max} - (k+1)$.

Definition 4. The set of all $(k+1)$ -pairs is the set $\Pi = \{\pi_1, \pi_2, \dots, \pi_M\}$, with $\langle m_1, m_2, z \rangle \in \Pi$ if and only if there exists some read pair $\langle R_1, R_2, z \rangle$ with m_1 a sub-molecule of R_1 and m_2 a sub-molecule of R_2 .

Note 3. When we allow duplicates, $M = O(N(l-k)^2)$, where N is the number of reads and l the maximum read length.

Definition 5. An **edge traversal** is a tuple of the form $t = \langle e, d \rangle$, with $e \in E$ and $d \in \{F, R\}$, with F corresponding to traversing the edge in the forward direction, and R in the reverse direction.

Definition 6. A **path** is a sequence of edge traversals: $T = \langle t_1, t_2, \dots, t_l \rangle$.

Note 4. In general, edges in the graph can be traversed multiple times, so there could exist t_i and t_j , $i \neq j$ and $e_i = e_j$. We always assume that paths being discussed are valid walks in the string graph, as described in the previous section.

Consider some $\pi_x = \langle m_{1x}, m_{2x}, z_x \rangle$ and traversal T . Let $\mathcal{L}_x = \{t_i | e_i = p(m_{1x}).e\}$ be the set of edge traversals in T to which m_{1x} maps. Let $\mathcal{R}_x = \{t_j | e_j = p(m_{2x}).e\}$ be the set of all edge traversals to which m_{2x} maps.

Definition 7. For each $(t_i, t_j) \in \mathcal{L}_x \times \mathcal{R}_x$, $i < j$, the **observed distance** of π_x is:

$$d(\pi_x, t_i, t_j) = \sum_{h=i+1}^{j-1} \|e_h\| + \sigma_i + \sigma_j$$

$$\sigma_i = \begin{cases} p(m_1).f & \text{if } d_i = R \\ \|\|p(m_1).e\| - p(m_1).f & \text{if } d_i = F \end{cases}$$

$$\sigma_j = \begin{cases} p(m_2).f & \text{if } d_j = F \\ \|\|p(m_2).e\| - p(m_2).f & \text{if } d_j = R \end{cases}$$

Definition 8. π_x **supports** T using t_i and t_j if and only if $\lfloor z_x \rfloor \leq d(\pi_x, t_i, t_j) \leq \lceil z_x \rceil$.

Definition 9. t_i and t_j are **supported by** Π if and only if there exists some π_x that supports the path using t_i and t_j .

Note 5. We call this support *weak* because the genomic distance between t_i and t_j can differ from the path distance by as much as $\lceil z_x \rceil - \lfloor z_x \rfloor$.

Definition 10. *The maximum distance expectation for t_i and t_j and some fragment type z , denoted by $\lceil (t_i, t_j, z) \rceil$, is calculated as $\min \left(\lceil z \rceil, \sum_{h=i}^j \|e_h\| \right)$.*

Definition 11. *The minimum distance expectation for t_i and t_j and some fragment type z , denoted by $\lfloor (t_i, t_j, z) \rfloor$, is calculated as $\max \left(\lfloor z \rfloor, \sum_{h=i+1}^{j-1} \|e_h\| \right)$.*

In general, multiple $(k+1)$ -pairs with the same fragment type can support a pair of edge traversals on a path. Moreover, if the path is correct, we would expect that, for all z_h , support for much of the range $\lfloor (t_i, t_j, z_h) \rfloor, \lceil (t_i, t_j, z_h) \rceil$ to be found in the data, assuming the edges are not very short. We wish to formalize this support expectation.

Definition 12. *A $(k+1)$ -pair cluster is a set of observed distances for t_i, t_j , and z . We summarize a cluster using the range $\alpha(t_i, t_j, z) = [min, max]$, with min being the minimum observed distance in the cluster and max the maximum observed distance.*

We construct the $(k+1)$ -pair clusters starting from all single element sets taken from Π and proceeding in two phases of merging. In the first phase, we perform single linkage clustering, merging two sets $\alpha_x(t_i, t_j, z)$ and $\alpha_y(t_i, t_j, z)$ if and only if $(max_x + R > min_y) \wedge (min_x - R < max_y)$, for some parameter R . In the second stage, we order all clusters by min , and then, considering all consecutive pairs $(\alpha_x(t_i, t_j, z), \alpha_y(t_i, t_j, z))$ in this ordered set, merge if $max_y - min_x < \lceil z \rceil$.

Definition 13. *t_i and t_j are strongly supported by a $(k+1)$ -pair cluster $\alpha(t_i, t_j, z)$ if $\alpha_{min} < \lfloor (t_i, t_j, z) \rfloor + T$ and $\alpha_{max} > \lceil (t_i, t_j, z) \rceil - T$, with T a sensitivity parameter.*

The preceding definition is carefully chosen to allow for an edge to be strongly supported even if, for example, t_i is a repeat and occurs at multiple distances from t_j in the genome. For a visual intuition behind the definition of strongly supported, see Figure 3.

In practice, we wish to be able to answer the question of whether t_i and t_j are strongly supported without having to consider the entire set Π when analyzing a particular path, but instead preprocess the raw paired reads to extract the necessary features. This needs to be done without any *a priori* knowledge of the nature of the eventual traversal T . In other words, we do not know either the distance between pairs of edges or their relative orientations at the time of summarization.

We achieve this goal by calculating, for each tuple $\langle e_i, e_j, z \rangle$, the ranges of the partial sum $\sigma_i + \sigma_j$ corresponding to each $(k+1)$ -pair cluster. As we do not know the relative orientation of the edges at the time of summation, we track the range of $\sigma_i + \sigma_j$ for all four possible orientations of edges, using ff_{min} and ff_{max} to denote this range when e_i and e_j are traversed forwards, with fr_{min} , fr_{max} , rf_{min} , rf_{max} , rr_{min} , and rr_{max} denoting the ranges of other orientations.

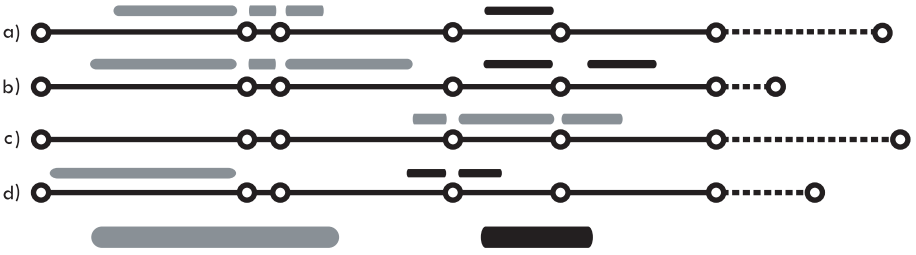


Fig. 3. Candidate path extensions, with the solid lines corresponding to a known path T and the dashed lines corresponding to some possible path extensions t'_j . The bars above each path correspond to cluster evidence supporting the extension of the edge, $\alpha(t_i, t'_j, z_h)$. Cluster ranges are truncated on the left such that they do not overlap between edges. The bars at the bottom of the figure correspond to the expected evidence for each fragment type, $[[z_h], [z_h]]$. In a), we show prototypical strong support for an extension across all edges. In b), we show support for a repeat that will occur twice in quick succession. In c), we show lack of support. Finally, in d) we show an example of lack of support, despite some overlap between clusters and the expected support range.

Definition 14. A **partial $(k + 1)$ -pair cluster** is a summarization of a set of observed partial sums $\sigma_i + \sigma_j$ for edges e_i, e_j and fragment type z , denoted as $\hat{\alpha}(e_i, e_j, z) = \langle rf_{min}, rf_{max}, ff_{min}, ff_{max} \rangle$.

Note 6. The value for rr_{min} can be calculated as $\|e_i\| + \|e_j\| - ff_{max}$, and rr_{max} , fr_{min} , and fr_{max} can be calculated similarly.

Given a traversal T with edges t_i and t_j , we calculate the $(k + 1)$ -pair clusters $\alpha_x(t_i, t_j, z)$ corresponding to partial $(k + 1)$ -pair cluster $\hat{\alpha}_y(e_i, e_j, z)$. If $t_i.f = F$ and $t_j.f = F$, $min_x = ff_{min_y} + \sum_{h=i+1}^{j-1} \|e_h\|$ and $max_x = ff_{max_y} + \sum_{h=i+1}^{j-1} \|e_h\|$. The other orientations of t_i and t_j are handled similarly.

We will now describe a parallel algorithm for computing all partial $(k + 1)$ -pair clusters from Π :

1. Assume that we have, for each $(k + 1)$ -molecule m , a mapping to that molecule's corresponding graph position, stored as a distributed tuple array of the form $\langle m, e_{ID}, f, l \rangle$, where l is the length of the edge identified by e_{ID} , and f is the forward position, as described previously.
2. Let C be the a distributed tuple array containing tuples of the form $\langle e_{ID_i}, e_{ID_j}, z, ff_{min}, ff_{max}, rf_{min}, rf_{max} \rangle$, corresponding to partial $(k + 1)$ -pair clusters. C is initially empty.
3. To reduce memory consumption, we process the paired reads in R stages. In each stage we process an $\frac{N}{R}$ subset of the data, $\frac{N}{Rp}$ per processor, if p is the number of processors.
 - (a) For each read pair in the data of the form $\langle R_1, R_2, z \rangle$, create $(k + 1)$ -molecule tuples $\langle m_i, m_j, z \rangle$ for all possible $(k + 1)$ -molecule combinations.
 - (b) Distribute these tuples, based on element m_i , to those processors that have the corresponding $(k + 1)$ -molecule tuple.

- (c) Combine the tuples $\langle m_i, m_j, z \rangle$ and $\langle m_i, e_{ID_i}, f_i, l_i \rangle$ to form the tuple $\langle m_j, z, e_{ID_i}, f_i, l_i \rangle$.
 - (d) Distribute the tuples, based on element m_j , to those processors that have the corresponding $(k + 1)$ -molecule tuple.
 - (e) Combine the tuples $\langle m_j, e_{ID_j}, f_j, l_j \rangle$ and $\langle m_j, z, e_{ID_i}, f_i, l_i \rangle$ to form a partial $(k + 1)$ -pair cluster $\langle e_{ID_i}, e_{ID_j}, z, \text{ff}_{min}, \text{ff}_{max}, \text{rf}_{min}, \text{rf}_{max} \rangle$ with $\text{ff}_{min} = \text{ff}_{max} = l_i - f_i + f_j$ and $\text{rf}_{min} = \text{rf}_{max} = f_i + f_j$.
 - (f) Merge these partial $(k + 1)$ -pairs with the array C .
 - (g) Sort C , using e_{ID_i} as primary key, e_{ID_j} as secondary key z as tertiary key, and finally by ff_{min} , forcing all clusters with equivalent (e_{ID_i}, e_{ID_j}, z) onto the same processor.
 - (h) For each bucket of partial $(k + 1)$ -pair clusters with equivalent (e_{ID_i}, e_{ID_j}, z) , merge partial $(k + 1)$ -pair clusters in accordance with the single linkage merging described above, updating all minimums and maximums. As the clusters are sorted by ff_{min} , this can be done using a single pass through the array on each processor.
4. After all stages have completed, consider each bucket of partial $(k + 1)$ -pair clusters with equivalent (e_{ID_i}, e_{ID_j}, z) , and merge partial $(k + 1)$ -pair clusters in accordance with the phase two merging rule described above, updating all minimums and maximums. As the clusters are sorted by ff_{min} , this can be done using a single pass through the array on each processor.
 5. Write the resulting partial clusters.

5 Bidirected Graph Traversal

Given the partial $(k + 1)$ -pair clusters, we wish to find valid walks through the bidirected string graph. We will describe the process using $(k + 1)$ -pair clusters, as it is more natural to do so, and these can be derived from the partial $(k + 1)$ -pair clusters for any path T .

We assign to each edge e in the graph an expected traversal bound $b(e)$ by analyzing the coverage seen along that edge. When traversing the graph, we keep track of the number of times an edge has been traversed as $c(e)$, and use this information in conjunction with the bound to choose between ambiguous options. Additionally, we restrict traversal to edges with $c(e) < 2b(e)$. Initially $c(e) = 0$ for all edges.

Our method for traversing the graph is one of path extension. Given a likely partial traversal of the graph as a path $T = \langle t_1, t_2, \dots, t_l \rangle$ with total length greater than the maximum fragment size, we can determine, by looking at the structure of the string graph, a set of possible edge traversals that can serve as extensions of this path: $E = \{t'_1, t'_2, \dots, t'_h\}$, $t'_j = \langle e_j, d_j \rangle$. We describe a heuristic method for choosing the best extension from E by choosing the candidate with the most strong support among the $(k + 1)$ -pair clusters.

Specifically, consider some extension t'_j . Let T' be the path created by extending T with t'_j .

Definition 15. *The expected support for t_i , t'_j , and fragment type z_v is:*

$$\gamma_{ijv} = \begin{cases} \hat{\gamma}_{ijv} & \text{if } (\lfloor (t_i, t'_j, z_v) \rfloor < (\lceil z_v \rceil - B)) \wedge (\lceil (t_i, t'_j, z_v) \rceil > (\lfloor z_v \rfloor + B)) \\ \hat{\gamma}_{ijv} & \text{if } t_i \text{ and } t'_j \text{ are strongly supported by some } \alpha(t_i, t'_j, z_v) \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{\gamma}_{ijv} = \lceil (t_i, t'_j, z_v) \rceil - \lfloor (t_i, t'_j, z_v) \rfloor$$

Where B is a parameter.

Definition 16. *The observed support for t_i , t'_j and fragment type z_v is:*

$$\omega_{ijv} = \begin{cases} \hat{\omega}_{ijv} & \text{if } t_i \text{ and } t'_j \text{ are strongly supported by some } \alpha(t_i, t'_j, z_v) \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{\omega}_{ijv} = \min(\lceil (t'_i, t'_j, z_v) \rceil, \alpha.max) - \max(\lfloor (t'_i, t'_j, z_v) \rfloor, \alpha.min)$$

Definition 17. *The extension score of a candidate t'_j (for $\sum_{i,v} \gamma_{ijv} > 0$) is defined as:*

$$0 \leq S(t'_j) = \frac{\sum_{i,v} \omega_{ijv}}{\sum_{i,v} \gamma_{ijv}} \leq 1$$

We say that an extension t'_j is *unambiguous* if $S(t'_j) > 0$ and, for all w with $0 < w \leq h$ and $w \neq j$, $\frac{S(t'_w)}{S(t'_j)} < D$ (for some specificity parameter D). If there exists an extension t'_j that is unambiguous, then we append t'_j to the path and continue with traversal. If t'_j does not exist, we consider only those edges with $b(t'_j) > c(t'_j)$, and look for the existence of an unambiguous extension \hat{t}'_j . If neither t'_j nor \hat{t}'_j exist, we stop extension.

All that remains is to describe how we seed the paths. All edges e with $\|e\|$ greater than the maximum fragment size and $c(e) = 0$ can serve as a seed. As it is not obvious, for example, that really long edges are better than moderately long edges as starting points, we simply choose from candidate starting points in increasing order of edge identifier, until none remain.

6 Experimental Results

We experimentally evaluated the proposed method using synthetic data generated from previously assembled genomes, downloaded from the NCBI FTP server. For each genome, we cleaned any ambiguities from the data and concatenated multiple contigs from the same chromosome, in the order presented in the finished FASTA file. In this way, we generate a contiguous sequence for each chromosome even though the actual data may contain a large number of contigs, possibly scaffolded. From the input chromosomes, we then generated fragments and sampled 30bp to 50bp paired short reads from the ends of the fragments.

Table 1. Assembly quality for five organisms. The first group shows results for sequences using protocol I, as described in the text. The second group was assembled from data matching protocol II. In order, we show the size of the genome in megabases; the maximum, $n50$, $n75$, and $n90$ lengths, all in kilobases; the number of contigs with length $> 10\text{Kb}$, the number of missassemblies per megabase, and percentage of the genome covered by these contigs at $> 99.9\%$ identity.

Organism	Size	Max	n50	n75	n90	Count	Mis	Cov
<i>E. coli</i>	5.4	224	85	43	10	91	0.2	90.0
<i>S. cerevisiae</i>	12.2	225	71	34	11	225	0.8	90.1
<i>C. pneumoniae</i>	1.0	867	867	867	132	2	0.0	99.9
<i>S. pneumoniae</i>	2.1	321	137	92	77	19	0.0	95.5
<i>E. coli</i>	5.4	378	231	104	42	42	0.5	94.0
<i>S. cerevisiae</i>	12.2	290	107	75	25	148	0.8	94.1
<i>D. melanogaster</i>	120.3	855	102	43	12	1,687	1.5	91.2

We used a 0.9% substitution rate and 0.1% deletion rate, for a total error rate of 1%. Fragment sizes were based upon two hypothetical experimental protocols. Protocol I consisted of two fragment types, $\{900 \pm 100, 4300 \pm 600\}$. Protocol II consisted of five fragment types $\{330 \pm 30, 660 \pm 60, 1100 \pm 100, 2200 \pm 200, 4400 \pm 400\}$. All genomes were sampled at 300-fold coverage.

For testing the assembler, data was generated on a workstation, using the parameters described above. This data was then transferred to a 512-node Blue Gene/L system with 512 MB memory per node, at which point the parallel phases of the software were run to produce the intermediate string graph and features from paired reads. These results were transferred back to a workstation for production of contigs using bidirected graph traversal. For *Drosophila*, this process took ~ 50 minutes for data transfer (depending on network congestion), ~ 100 minutes for the parallel phases, and ~ 20 minutes for the remaining. In the final processing phase of the algorithm, run time was dominated by file I/O.

Producing long and correctly assembled contigs that cover most of the genome is the hallmark of a good assembler. We use the nX length measure, which is the maximum length l such that X percent of the genome is covered by contigs with length at least l . For validation, we used MUMmer 3.20 [12] to align the finished contigs back to the reference. The results are presented in Table 1. We present results on three bacterial genomes as a reference point for comparison with other work on short read assembly. In addition, we present results on *S. cerevisiae* and *D. melanogaster*. For each genome, we present the length of the maximum contig generated, along with $n50$, $n75$ and $n90$ lengths. We also present the number of contigs with length $> 10\text{Kb}$, the percentage of the genome that is covered by these contigs with at least 99.9% identity (typically, four out of five assembled contigs aligned perfectly), and the number of large-scale missassemblies per megabase.

7 Discussion and Conclusions

In this paper, we presented a graph-based method for assembly of large genomes from short reads. Our method can scale to large genomes using distributed memory multiprocessors. It can naturally handle reads of multiple lengths (short reads from one or more platforms, Sanger reads, or a mixture), and multiple fragment sizes used in paired read generation. Experimental results show that our method produces large ($n50 > 100\text{Kb}$), high quality ($>99.9\%$ correct) contigs from synthetic data sets generated with 1% error in a few hours of wall clock time using a 512-node Blue Gene/L. We have validated our method by generating synthetic short read data from drosophila, yeast, and a number of bacterial genomes.

Error discovery is an essential part of any *de novo* short read assembly. While we have identified errors by counting k -mer frequency for relatively large k , other promising methods for finding errors have been proposed. Zerbino *et al.* [23] describe motifs in the graph that are likely to be due to errors, such as alternate edges between two nodes, with one edge at much lower coverage. Chaisson *et al.* [4] describe methods for preserving sequences with errors by finding good edits. We are interested in exploring ways to achieve robust error correction using parallel computers, as this will be necessary to achieve good results at lower sequence coverage rates.

There is good potential for using the large edges in the graph to decompose the graph traversal into multiple independent problems. We plan to develop parallel algorithms for this final assembly stage, leading to the creation of a fully parallel assembly pipeline. This may be necessary for scaling to very large genomes such as human, mouse and maize. Using our assembler, it has become easy to experiment with the number and types of paired read lengths. A parameterized study with multiple experimental protocols, and different sampling errors and coverage variance, can be conducted to help plan future *de novo* genome sequencing projects.

Acknowledgements

We thank Chad Brewbaker, Scott Emrich, Xiao Yang, and Jaroslaw Zola for their input and feedback. This project was supported in part by the National Science Foundation under CNS-0521568 and by the Plant Sciences Institute Innovative Research Grants program.

References

1. Bennet, S.: Solexa ltd. *Pharmacogenomics* 5(4), 433–438 (2004)
2. Bentley, D.R., Balasubramanian, S., Swerdlow, H.P., Smith, G.P.: Accurate whole human genome sequencing using reversible terminator chemistry. *Nature* 456, 53–59 (2008)
3. Butler, J., MacCallum, I., Kleber, M., Shlyakhter, I.A., Belmonte, M.K., Lander, E.S., Nusbaum, C.N., Jaffe, D.B.: ALLPATHS: De novo assembly of whole-genome shotgun microreads. *Genome Research* 18, 810–820 (2008)

4. Chaisson, M.J., Pevzner, P.A.: Short fragment assembly of bacterial genomes. *Genome Research* 18, 324–330 (2008)
5. Dohm, J.C., Lottaz, C., Borodina, T., Himmelbauer, H.: SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. *Genome Research* 17, 1697–1706 (2007)
6. Helman, D.R., Ja'Ja', J., Bader, D.A.: A new deterministic parallel sorting algorithm with an experimental evaluation. *Journal of Experimental Algorithms* 3, 4 (1998)
7. Hernandez, D., Francois, P., Farinelli, L., Osteras, M., Schrenzel, J.: De novo bacterial genome sequencing: Millions of very short reads assembled on a desktop computer. *Genome Research* 18, 802–809 (2008)
8. Hossain, S., Azimi, N., Skiena, S.: Crystallizing short-read assemblies around lone Sanger reads. *Bioinformatics* (2009)
9. Idury, R.M., Waterman, M.S.: A new algorithm for DNA sequence assembly. *Journal of Computational Biology* 2, 291–306 (1995)
10. Jackson, B.G., Aluru, S.: Parallel construction of bidirected string graphs for genome assembly. In: *Proceedings of the International Conference on Parallel Processing*, pp. 346–353 (2008)
11. Jackson, B.G., Schanble, P.S., Aluru, S.: Parallel short sequence assembly of transcriptomes. *BMC Bioinformatics* 10, S14 (2009)
12. Kurtz, S., Phillippy, A., Delcher, A.L., Smoot, M., Shumway, M., Antonescu, C., Salzberg, S.L.: Versatile and open software for comparing large genomes. *Genome Biology* 5 (2004)
13. Margulies, M., Egholm, M.: Genome sequencing in open microfabricated high density picoliter reactors. *Nature* 437(7054), 376–380 (2005)
14. Medvedev, P., Brudno, M.: Ab initio whole genome shotgun assembly with mated short reads. In: Vingron, M., Wong, L. (eds.) *RECOMB 2008*. LNCS (LNBI), vol. 4955, pp. 50–64. Springer, Heidelberg (2008)
15. Myers, E.W.: The fragment assembly string graph. *Bioinformatics* 21, ii79–ii85 (2005)
16. Ossowski, S., Schneeberger, K., Clark, R.M., Lanz, C., Warthmann, N., Weigel, D.: Sequencing of natural strains of *Arabidopsis thaliana* with short reads. *Genome Research* (preprint) (2008)
17. Pandey, V., Nutter, R.C., Prediger, E.: *Applied Biosystems SOLiD System: Ligation-Based Sequencing*. Wiley, Chichester (2008)
18. Pevzner, P.A., Tang, H., Waterman, M.S.: Fragment assembly with double-barreled data. *Proceedings of the National Academy of Sciences* 98(17), 9748–9753 (2001)
19. Wang, J., Wang, W., Li, R., Li, Y.: The diploid genome sequence of an Asian individual. *Nature* 456, 60–65 (2008)
20. Warren, R.L., Sutton, G.G., Jones, S.J.M., Holt, R.A.: Assembling millions of short DNA sequences using SSAKE. *Bioinformatics* 23, 500–501 (2007)
21. Wicker, T., Narechania, A., Sabot, F., Vu, G.T.H., Graner, A., Ware, D., Stein, N.: Low-pass shotgun sequencing of the barley genome facilitates rapid identification of genes, conserved non-coding sequences and novel repeats. *BMC Genomics* 9, 518 (2008)
22. Business Wire. Helicos biosciences enters molecular diagnostics collaboration with renowned research center to sequence cancer-associated genes. *Genetic Engineering and Biotechnology News* (2008)
23. Zerbino, D., Birney, E.: Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research* 18, 821–829 (2008)

Amino Acid Classification and Hash Seeds for Homology Search

Weiming Li^{1,*}, Bin Ma^{2,**}, and Kaizhong Zhang^{1,***}

¹ Department of Computer Science, University of Western Ontario,
London, ON, Canada N6A 5B7
{wli5,kzhang}@csd.uwo.ca

² School of Computer Science, University of Waterloo,
Waterloo, ON, Canada N2L 3G1
binma@uwaterloo.ca

Abstract. Spaced seeds have been extensively studied in the homology search field. A spaced seed can be regarded as a very special type of hash function on k -mers, where two k -mers have the same hash value if and only if they are identical at the w ($w < k$) positions designated by the seed. Spaced seeds substantially increased the homology search sensitivity. It is then a natural question to ask whether there is a better hash function (called *hash seed*) that provides better sensitivity than the spaced seed. We study this question in the paper. We propose a strategy to classify amino acids, which leads to a better hash seed. Our results raise a new question about how to design the best hash seed.

1 Introduction

The homology search aims to find the approximate local matches between two DNA or protein sequences (or sequence databases). Since the large sizes of the DNA and protein sequence databases, a “filtration” is usually applied to rapidly identify the potential matching locations and then a more accurate but time-consuming examination is done on the identified locations. Among the first to adopt this approach in homology search area, the famous BLAST program [1] uses the exact matches of w -mers (called *hits*) as the criterion of the filtration. Then a more careful examination that includes the Smith-Waterman algorithm [31] is used to further examine those hitting locations. Clearly, this speeds up the homology search as w -mer matches are much easier to identify than approximate matches. On the other hand, the sensitivity of the homology search is sacrificed because a highly similar region does not necessarily contain a w -mer match and then gets lost.

The PatternHunter paper [27] introduced the optimized spaced seed method to increase the sensitivity. A length- k , weight- w spaced seed is w designated

* Supported by NSERC.

** Supported by an NSERC grant, a start up grant at University of Waterloo, and China high-tech R&D program (863) 2008AA02Z313.

*** Supported by NSERC.

positions out of k . A spaced seed can be either given by a set of these w positions (e.g. $\{1,2,4\}$ for a weight-3, length-4 seed), or equivalently, a binary string with only those selected positions equal to 1 (e.g. 1101 for the spaced seed given above). Two k -mers are *hit* by the seed if the two k -mers match at the w designated positions. The PatternHunter program uses these hits to identify potential homologies and then examine the hitting locations similarly to BLAST. Surprisingly, an optimized spaced seed was demonstrated in [27] and mathematically proved in [18,24,9] to have much higher sensitivity than using w -mers. Outside of the context of homology search, some earlier works [29,22] prior to PatternHunter also proposed to use some discontinuous positions as a filter for similarity search, without attempting to optimize these positions.

The discovery of optimized spaced seed provoked significant amount of researches in optimizing the spaced seed method. These include the modifications to the original spaced seed method (see, e.g., [5,23,4,21,12,13]), the algorithms and complexities to compute seed sensitivities and to optimize spaced seeds and modified spaced seeds (see, e.g., [28,24,18,26,16,33,17,19,9,11,32,20,10]), and the use of spaced seeds in different applications (see, e.g., [2,6,25]). The readers are referred to the review articles on spaced seed method [8,7] for more details.

Among the modifications to the original spaced seed method, the *multiple spaced seeds* [23] use several spaced seeds simultaneously to improve the sensitivity-speed tradeoff. The vector seeds [3,5] allow the w designated positions in a spaced seed to be inexact matches. A threshold is predefined to check whether the inexact match is “good enough” to be a hit. Vector seeds can be regarded as a combination and extension of the spaced seed idea with the neighborhood idea used in blastp (a subprogram of BLAST) or BLASTZ [30]. Brown further designed multiple vector seeds [6] specialized for protein homology search. The neighbor seeds [13] select 1 or 2 of the w positions to allow them to be mismatches, but at the same time require 1 or 2 additional positions outside the w positions to be matches. This allows to use slightly different spaced seeds to perform the homology search even after the index table is built using a given spaced seed.

All these modifications require the change of either the indexing algorithm or the hits generation algorithm. For multiple spaced seeds, multile index tables, each for a spaced seed, need to be built. This greatly increases the space complexity. For the vector seeds, the w positions of a k -mer of the query sequence are regarded as a w -dimension vector, and the “neighbors” within certain distance to the vector are generated and used for the look up in the index table. Therefore, multiple queries to the index table are needed for each single location of the query sequence. This adds overhead to the hits generation, especially if the hash table is to be stored on secondary storage media. Consequently, only parameters that cause not too many neighbors are allowed in practical vector seeds design. For the neighbor seeds, not only the hits generation, but also the process after a hit is found needs to be changed. The overhead for these changes is the reason why only 1 or 2 mismatches are used in neighbor seeds.

It was pointed out in [13], and probably widely known, that a weight- w , length- k , spaced seed is just a special hash function for k -mers. Two k -mers have

the same hash value if and only if they are identical at the w designated positions of the given spaced seed. From this point of view, if a different hash function is used instead of the spaced seed, the original PatternHunter’s algorithm does not require any changes and no overhead is added. The paper [13] acclaimed that the spaced seeds just happen to be a simple but good hash functions in terms of sensitivity-speed tradeoff. For the sake of clarity, we call a k -mer hash function, when used in lieu of a spaced seed, as a *hash seed*.

Spaced seeds are only a very special type of hash seeds. It is natural to believe that there are better hash seeds than spaced seeds for the homology search purpose. However, for general homology search, no better hash seed has been found to date. In this paper we design such a hash seed for protein homology search. The hash seed provides noticeably better sensitivity than optimized spaced seed, at the same false positive level (and therefore same speed). Our contribution in the paper raises the question of finding the optimal hash seed with the best sensitivity-speed trade off.

The rest of the paper is organized as follows: Section 2 introduces the idea of classifying amino acids according to a BLOSUM matrix. Section 3 employs the amino acid classification to give a simple hash seed. Section 4 discusses the results and future work.

2 Classification of Amino Acids

The homology search of protein sequences usually uses the BLOSUM matrices to measure the similarities between amino acids. BLOSUM matrices was introduced by Henikoff and Henikoff [14]. The BLOSUM62 matrix was recommended for the BLAST program. Figure 1 shows the BLOSUM62 matrix.

Note that in Figure 1, we rearranged the amino acids so that the positive scores are concentrated around the diagonal of the matrix. As a result, it becomes apparent that some amino acids with positive scores to each other can be classified together. Two classification schemes are shown in Figure 1. For the two classification schemes, the minimum score within a class is 2 and 1, respectively. 1 We call these two classifications as \mathcal{C}^2 and \mathcal{C}^1 , respectively. For the sake of clarity, we also call the trivial classification that puts each amino acid in a different class as \mathcal{C}^* .

Let Σ be the alphabet of all amino acids. A classification is defined as $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$. Each C_i is a subset of Σ that satisfies $C_i \cap C_j = \emptyset$ for $i \neq j$, and $\bigcup_{i=1}^k C_i = \Sigma$. We use $|\mathcal{C}| = k$ to denote the number of different classes of the classification, and use $\mathcal{C}(a)$ to denote the class that a belongs to. The integer i is called the class id of C_i or a letter $a \in C_i$.

Although the BLOSUM matrices were obtained purely statistically [14], we notice that the classifications given in Figure 1 make very much biological sense. For example, both D and E are polar, negatively charged, non-hydrophobic

¹ In the second classification scheme, it is possible to additionally put N and H into a new class. However, this has only negligible effects to our results.

	C	G	A	T	S	N	D	E	Q	K	R	V	I	L	M	W	F	Y	H	P
C	9	-3	0	-1	-1	-3	-3	-4	-3	-3	-1	-1	-1	-1	-2	-2	-2	-3	-3	
G	-3	6	0	-2	0	0	-1	-2	-2	-2	-2	-3	-4	-4	-3	-2	-3	-3	-2	-2
A	0	0	4	0	1	-2	-2	-1	-1	-1	0	-1	-1	-1	-3	-2	-2	-2	-1	
T	-1	-2	0	5	1	0	-1	-1	-1	-1	0	-1	-1	-1	-2	-2	-2	-1		
S	-1	0	1	1	4	1	0	0	0	0	-1	-2	-2	-2	-1	-3	-2	-1	-1	
N	-3	0	-2	0	1	6	1	0	0	0	0	-3	-3	-3	-2	-4	-3	1	-2	
D	-3	-1	-2	-1	0	1	6	2	0	-1	-2	-3	-3	-4	-3	-4	-3	-1	-1	
E	-4	-2	-1	-1	0	0	2	5	2	1	0	-2	-3	-3	-2	-3	-2	0	-1	
Q	-3	-2	-1	-1	0	0	0	2	5	1	1	-2	-3	-2	0	-2	-3	-1	0	-1
K	-3	-2	-1	-1	0	0	-1	1	5	2		-2	-3	-2	-1	-3	-3	-2	-1	-1
R	-3	-2	-1	-1	-1	0	-2	0	1	2	5	-3	-3	-2	-1	-3	-3	0	-2	
V	-1	-3	0	0	-2	-3	-3	-2	-2	-2	-3	4	3	1	1	-3	-1	-1	-3	-2
I	-1	-4	-1	-1	-2	-3	-3	-3	-3	-3	-3	3	4	2	1	-3	0	-1	-3	-3
L	-1	-4	-1	-1	-2	-3	-4	-3	-2	-2	-2	1	2	4	2	-2	0	-1	-3	-3
M	-1	-3	-1	-1	-1	-2	-3	-2	0	-1	-1	1	1	2	5	-1	0	-1	-2	-2
W	-2	-2	-3	-2	-3	-4	-4	-3	-2	-3	-3	-3	-3	-2	-1	11	1	2	-2	-4
F	-2	-3	-2	-2	-2	-3	-3	-3	-3	-3	-3	-1	0	0	0	1	6	3	-1	-4
Y	-2	-3	-2	-2	-2	-3	-2	-1	-2	-2	-1	-1	-1	-1	2	3	7	2	-3	
H	-3	-2	-2	-2	-1	1	-1	0	0	-1	0	-3	-3	-3	-2	-2	-1	2	8	-2
P	-3	-2	-1	-1	-1	-2	-1	-1	-1	-2	-2	-3	-3	-2	-4	-4	-3	-2	7	

Fig. 1. The BLOSUM62 matrix. The highlighted blocks show an exemplary classification such that the score within each class is at least 2. The blocks surrounded by rectangles show an exemplary classification such that the score within each class is at least 1.

amino acids, and they are classified together in Figure 1. However, because the scores in the BLOSUM matrices are greatly affected by the amino acid frequencies (which do not necessarily reflect the chemical properties of the amino acids), and were rounded to integers, our classifications based on BLOSUM62 matrix may not accurately reflect a classification solely using amino acids’ chemical properties.

Clearly, in the sequence alignment of a homology, if we replace each amino acid with its class id, the identity level of the alignment will increase. This will increase the sensitivity of the homology search if the same spaced seed is used. However, this will also increase the identity level of the match of two random sequences, resulting into more random hits and slower search speed. We need to make sure that we gain more from the sensitivity than losing at the speed. This tradeoff is discussed in greater detail in next section.

3 A Hash Seed Based on Amino Acid Classification

In [14], the score in a BLOSUM matrix was calculated as follows. First, for two amino acids x and y ,

$$sc(x, y) = 2 \log_2 \frac{p_{xy}}{2p_x \cdot p_y},$$

where p_{xy} is the (normalized) frequency of observing x and y aligned together in a real homolog, and p_x and p_y are the background frequencies of x and y in the protein database. Then this $sc(x, y)$ is rounded to the nearest integer to create the BLOSUM matrix. Therefore, a positive score between two amino acids indicate that the two amino acids match each other more often in homologies than they would in random sequence matches. Consequently, if we only classify positive scoring amino acid pairs together, we will get more hits at the real homologies than at the random matched sequences. Thus, one idea of designing a good hash seed is to classify the amino acids and use spaced seeds on sequences of amino acid classes.

More accurately, let $\{i_1, i_2, \dots, i_w\} \subseteq [1, k]$ be a spaced seed and $s = a_1 a_2 \dots a_k$ be a k -mer. Let \mathcal{C} be an amino acid classification. Then our hash function h is defined as [2](#)

$$h(s) = \mathcal{C}(a_{i_1})\mathcal{C}(a_{i_2}) \dots \mathcal{C}(a_{i_w}). \quad (1)$$

Similarly to the spaced seed, two k -mers generate a hit if and only if they have the same hash value based on our new hash function.

The *false positive rate* of a hash seed is defined to be the probability that two random k -mers have the same hash value. Clearly, a smaller false positive rate results into higher search speed. Let F_p be the false positive rate, we also refer to $\frac{1}{F_p}$ as the *selectivity* of the hash seed. Let p_a be the background frequency of an amino acid a . The F_p of the hash seed in Eq. [\(1\)](#) can be calculated easily as

$$F_p(h) = \left(\sum_{i=1}^{|\mathcal{C}|} \left(\sum_{a \in C_i} p_a \right)^2 \right)^w \quad (2)$$

The *sensitivity* of the hash seed is calculated by using the real data as follows. We used Arabidopsis Chromosome 2 and Chromosome 4 as our testing data. The protein sequences from the two chromosomes are downloaded from NCBI's website and put in two separate FASTA format files. The SSearch program [\[15\]](#) was used to find all the high-scoring local alignments between the two FASTA files. BLOSUM62 was used as SSearch's scoring matrix. Then each alignment was splitted into ungapped alignments at the insertion and deletion locations. These ungapped alignments are called high-scoring segment pairs (HSPs). The score of each HSP is recalculated using BLOSUM62. There are 142790 HSPs with scores greater than or equal to 50. We use the percentage of these HSPs that generate hits as the sensitivity of the (hash) seed. The p_a that is used to compute the F_p in Eq. [\(2\)](#) is also calculated from these two FASTA files.

By using seeds of different weights, we can trade between sensitivity and selectivity. Figure [2](#) show the performance of the hash seed h defined in Eq. [\(1\)](#) for classification \mathcal{C}^1 , \mathcal{C}^2 , and \mathcal{C}^* , respectively. Note that the hash seed for \mathcal{C}^* is equivalent to the spaced seed.

² Note that the hash function in Eq. [\(1\)](#) results into a string of amino acid classes instead of an integer. This can be easily amended by assigning each distinct string of amino acid classes with a unique integer. For this reason, we will not require a hash function to have integer values throughout the paper.

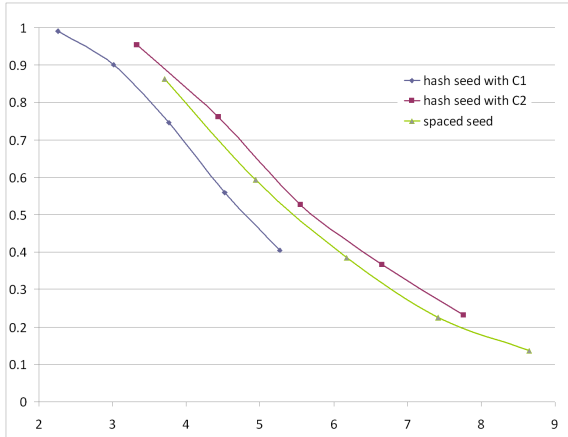


Fig. 2. The sensitivity-selectivity trade off for different hash seeds. Each data point indicate a hash seed defined in Eq. (II) using a weight- w seed and an amino acid classification \mathcal{C}^1 , \mathcal{C}^2 , or \mathcal{C}^* . The x-axis is $\log_{10}(\text{selectivity})$. The y-axis is the sensitivity of the hash seed. The sensitivity is calculated using the HSPs with scores greater than or equal to 50.

From the figure we can see that the hash seed with classification \mathcal{C}^2 have a better sensitivity-selectivity tradeoff than the spaced seed. Whereas the hash seed with classification \mathcal{C}^1 has worse performance than spaced seed.

4 Discussion

We presented a hash seed strategy for better sensitivity-speed tradeoff in homology search. We demonstrated that for protein homology search, where the alphabet letters are distributed very unevenly, it is possible to design a hash seed that is better than the optimized spaced seed. Our contribution raised the interesting question of how to design the best hash seed.

One possibility is to design hash seed by classifying k -mers. Any standard classification algorithm can be used. For example, a straightforward algorithm is to greedily group the highest-scoring k -mer pairs together. The authors are currently examining this possibility.

The hash seed idea can possibly be combined with the vector seed idea to further improve the sensitivity. But the exact outcome of this combination is out of this paper's scope and needs to be examined in future research.

Our designed hash seed takes advantage of the uneven distribution of the 20 different amino acids in protein sequences and alignments. Therefore, our design does not straightforwardly apply to the situation when the alphabet letters are uniformly distributed, a situation that resembles the DNA sequences. In fact, the authors moderately believe that under uniform distribution, the optimized spaced seed is the optimal hash seed.

References

1. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic local alignment search tool. *J. Mol. Biol.* 215(3), 403–410 (1990)
2. Brejová, B., Brown, D.G., Vinař, T.: Optimal spaced seeds for hidden markov models, with application to homologous coding regions. In: Baeza-Yates, R., Chávez, E., Crochemore, M. (eds.) *CPM 2003. LNCS*, vol. 2676, pp. 42–54. Springer, Heidelberg (2003)
3. Brejová, B., Brown, D., Vinař, T.: Vector seeds: An extension to spaced seeds allows substantial improvements in sensitivity and specificity. In: Benson, G., Page, R.D.M. (eds.) *WABI 2003. LNCS (LNBI)*, vol. 2812, pp. 39–54. Springer, Heidelberg (2003)
4. Brejová, B., Brown, D., Vinař, T.: Optimal spaced seeds for homologous coding regions. *J. Bioinf. and Comp. Biol.* 1(4), 595–610 (2004); early version appeared in: Baeza-Yates, R., Chávez, E., Crochemore, M. (eds.) *CPM 2003. LNCS*, vol. 2676, pp. 42–54. Springer, Heidelberg (2003)
5. Brejova, B., Brown, D.G., Vinar, T.: Vector seeds: an extension to spaced seeds. *Journal of Computer and System Sciences* 70(3), 364–380 (2005); early version appeared in: Benson, G., Page, R.D.M. (eds.) *WABI 2003. LNCS (LNBI)*, vol. 2812, pp. 39–54. Springer, Heidelberg (2003)
6. Brown, D.: Multiple vector seeds for protein alignment. In: Jonassen, I., Kim, J. (eds.) *WABI 2004. LNCS (LNBI)*, vol. 3240, pp. 170–181. Springer, Heidelberg (2004)
7. Brown, D.: A survey of seeding for sequence alignments. In: *Bioinformatics Algorithms: Techniques and Applications*, pp. 117–142. Wiley, Chichester (2008)
8. Brown, D.G., Li, M., Ma, B.: A tutorial of recent developments in the seeding of local alignment. *Journal of Bioinformatics and Computational Biology* 2(4), 819–842 (2004)
9. Buhler, J., Keich, U., Sun, Y.: Designing seeds for similarity search in genomic DNA. In: *Proc. of the 7th International Conference on Computational Biology (RECOMB)*, pp. 67–75 (2003)
10. Choi, K.P., Zeng, F., Zhang, L.: Good spaced seeds for homology search. *Bioinformatics* 20, 1053–1059 (2004)
11. Choi, K.P., Zhang, L.: Sensitive analysis and efficient method for identifying optimal spaced seeds. *J. Comp and Sys. Sci* 68, 22–40 (2004)
12. Csűrös, M.: Performing local similarity searches with variable length seeds. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) *CPM 2004. LNCS*, vol. 3109, pp. 373–387. Springer, Heidelberg (2004)
13. Csűrös, M., Ma, B.: Rapid homology search with neighbour seeds. *Algorithmica* 48(2), 187–202 (2007)
14. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. *Proc. of the National Academy of Sciences of the United States of America* 89(22), 10915–10919 (1992)
15. Huang, X., Hardison, R.C., Miller, W.: A space-efficient algorithm for local similarities. *CABIOS* 6, 373–381 (1990)
16. Yang, I.H., Wang, S.H., Chen, H.H., Huang, P.H., Chao, K.M.: Efficient methods for generating optimal single and multiple spaced seeds. In: *Proc. of IEEE 4th Symp. on Bioinformatics and Bioengineering*, pp. 411–418 (2004)
17. Ilie, L., Ilie, S.: Fast computation of good multiple spaced seeds. In: *Proc. of 7th Workshop on Algorithms in Bioinformatics* (2007)

18. Keich, U., Li, M., Ma, B., Tromp, J.: On spaced seeds for similarity search. *Discrete Appl. Math.* 3, 253–263 (2004)
19. Keich, U., Li, M., Ma, B., Tromp, J.: On Spaced Seeds of similarity search. *Discrete Appl. Math.* 138, 253–263 (2004)
20. Kucherov, G., Noe, L., Ponty, Y.: Estimating seed sensitivity on homogeneous alignments. In: Proc. of the 4th IEEE Symposium on Bioinformatics and Bioengineering (BIBE), pp. 387–394 (2004)
21. Kucherov, G., Noe, L., Roytberg, M.: Multi-seed lossless filtration. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) CPM 2004. LNCS, vol. 3109, pp. 297–310. Springer, Heidelberg (2004)
22. Lehtinen, O., Sutinen, E., Tarhio, J.: Experiments on block indexing. In: Proc. of the 3rd South American Workshop on String Processing (1996)
23. Li, M., Ma, B., Kisman, D., Tromp, J.: PatternHunter II: Highly sensitive and fast homology search. *J. Bioinf. and Comp. Biol.* 2(3), 417–440 (2004)
24. Li, M., Ma, B., Zhang, L.: Superiority and complexity of the spaced seeds. In: Proc. of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 22–26 (2006)
25. Lin, H., Zhang, Z., Zhang, M., Ma, B., Li, M.: Zoom! zillions of oligos mapped. *Bioinformatics* 24(21), 2431–2437 (2008)
26. Ma, B., Li, M.: On the complexity of spaced seeds. *Journal of Computer Science and System Sciences* (2007)
27. Ma, B., Tromp, J., Li, M.: PatternHunter: faster and more sensitive homology search. *Bioinformatics* 18(3), 440–445 (2002)
28. Nicolas, F., Rivals, E.: Hardness of optimal spaced seed design. In: Apostolico, A., Crochemore, M., Park, K. (eds.) CPM 2005. LNCS, vol. 3537, pp. 144–155. Springer, Heidelberg (2005)
29. Pevzner, P.A., Waterman, M.S.: Multiple filtration and approximate pattern matching. *Algorithmica* 13, 135–154 (1995)
30. Schwartz, S., Kent, W.J., Smit, A., Zhang, Z., Baertsch, R., Hardison, R.C., Hausler, D., Miller, W.: Human-mouse alignments with BLASTZ. *Genome Research* 13, 103–107 (2003)
31. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. *J. Mol. Biol.* 147, 195–197 (1981)
32. Sun, Y., Buhler, J.: Designing multiple simultaneous seeds for dna similarity search. In: Proc. of the 8th International Conference on Computational Biology (RECOMB), pp. 76–84 (2004)
33. Xu, J., Brown, D., Li, M., Ma, B.: Optimizing multiple spaced seeds for homology search. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) CPM 2004. LNCS, vol. 3109, pp. 47–58. Springer, Heidelberg (2004)

Genotype and Haplotype Reconstruction from Low-Coverage Short Sequencing Reads*

Ion Măndoiu

Department of Computer Science & Engineering, University of Connecticut, 371
Fairfield Rd., Unit 2155, Storrs, CT 06269-2155, USA
`ion@engr.uconn.edu`

Recent advances in *high-throughput sequencing* (HTS) technologies have led to orders of magnitude higher throughput compared to classic Sanger sequencing (see [3] for a review). Coupled with continuously decreasing sequencing costs, HTS data provides opportunities to study genome structure, function, and evolution at an unprecedented scale, and is profoundly transforming biomedical research.

One of the most exciting developments in this area is the advent of individual genome sequencing. Indeed, several individual genomes have already been published [2,4,7,8], and there are ongoing efforts to sequence thousands more [1]. Although technologies such as SNP arrays remain instrumental in genome-wide association studies that have already led to the discovery of hundreds of genes associated with common human diseases [5], sequencing provides a much more complete picture of genetic variation. The ideal outcome of an individual sequencing project is a diploid genome – i.e., full haplotype sequences for the individual’s maternal and paternal chromosomes – since only haplotype sequences provide the detailed context required for accurate *functional characterization* of genomic variants [6] and studying genome evolution. However, realizing this ideal requires the development of novel computational methods capable of reconstructing full haplotypes from HTS reads that are much shorter and include a significantly larger percentage of errors compared to Sanger sequencing reads.

This talk will outline computationally efficient algorithms that allow accurate multi-locus genotype and haplotype reconstruction by integrating HTS data with linkage disequilibrium (LD) information extracted from a reference population panel such as Hapmap and present preliminary results on publicly available 454, Illumina, and ABI SOLiD sequencing datasets. We will also discuss future prospects and remaining challenges in medical sequencing.

This is joint work with S. Dinakar, J. Duitama, Y. Hernandez, J. Kennedy, and Y. Wu.

References

1. 1000 Genomes Project Consortium, <http://www.1000genomes.org/>
2. Bentley, D.R., et al.: Accurate whole human genome sequencing using reversible terminator chemistry. *Nature* 456(7218), 53–59 (2008)

* Work supported in part by NSF awards IIS-0546457 and DBI-0543365.

3. Holt, R.A., Jones, S.J.: The new paradigm of flow cell sequencing. *Genome Res.* 18(6), 839–846 (2008)
4. Levy, S., et al.: The diploid genome sequence of an individual human. *PLoS Biology* 5(10), e254+ (2007)
5. Manolio, T.A., Brooks, L.D., Collins, F.S.: A HapMap harvest of insights into the genetics of common disease. *J. Clin. Invest.* 118(5), 1590–1605 (2008)
6. Ng, P.C., et al.: Genetic variation in an individual human exome. *PLoS Genet* 4(8), e1000160+ (2008)
7. Wang, J., et al.: The diploid genome sequence of an Asian individual. *Nature* 456(7218), 60–65 (2008)
8. Wheeler, D.A., et al.: The complete genome of an individual by massively parallel DNA sequencing. *Nature* 452, 872–876 (2008)

Gene Networks Viewed through Two Models

Satoru Miyano, Rui Yamaguchi, Yoshinori Tamada, Masao Nagasaki,
and Seiya Imoto

Human Genome Center, Institute of Medical Science, University of Tokyo
4-6-1 Shirokanedai, Minatoku, Tokyo 108-8639, Japan
miyano@ims.u-tokyo.ac.jp

Abstract. This paper presents our computational and measurement strategy for investigating gene networks from gene expression data using state space model and dynamic Bayesian network model with nonparametric regression. These methods are applied to gene expression data based on gene knockdowns and drug responses for generating large global maps of gene regulation which will light up the geography where drug target pathways lie down.

1 Introduction

Mining molecular networks from gene expression data, protein-protein interaction data and some other high-throughput data is an important computational challenge in Systems Biology. In the last decade, this challenge has received tremendous attentions and very strong computational methods have been developed from various viewpoints such as methodology, quality of data, data types, measurement devices, purpose of applications, computational resources, etc. The amount of contributions on this topic exceeds the space limit of this paper to survey them even shortly. The impact of this challenge on gene networks is being penetrating into biology and its related fields, especially, pharmacogenomics.

We have developed a series of computational methods for inferring gene networks and developed a software tool for visualizing and analyzing the molecular networks, e.g., Cell Illustrator Online (<http://cionline.hgc.jp/>). For the development of computational methods, we used convenient “public data” which were produced for some biological analyses and are independent of the computational methodology. In parallel to public data, we have been involved with biological experiment design and production of gene expression data whose purpose is to build large gene networks from gene expression data based on gene knockdowns and drug responses [14, 8, 15, 16]. Actually, we first succeeded in developing human gene networks of size more than 1000 from several hundreds gene knockdowns and drug responses in time course [8].

The purpose of this paper is to show some cases from our work that biological measurement experiments were designed in parallel to the development of computational methods for gene networks so that the computational methods and data will be harmonized. We present two modeling methods to view gene networks from gene expression data. One is state space model that has been used

in a wide variety of areas for modeling time-course data and dynamic systems in sciences other than biology [10]. Due to the characteristics of biological measurement experiments, there lie difficulties to overcome. The other is Bayesian network with nonparametric regression [5,6]. Bayesian network for discrete random variables was first applied for analyzing the gene network controlling cell cycles of *S. cerevisiae* by Friedman et al. [3]. However, gene expression data are continuous and relationships between gene expressions are sometimes nonlinear. Thus we constructed a framework that combine Bayesian network and nonparametric regression with B -splines and derived an information criterion called BNRC for searching the best gene networks. Through these two models, gene networks of HUVEC (human umbilical vein endothelial cell) and SAEC (normal human small airway epithelial cell) were analyzed against drugs by using the gene expression data produced for gene networks.

Our conclusive message on gene network inference is obviously simple that biological measurement experiments should be designed and conducted in accordance with computational methodology.

2 State Space Model for Modeling Gene Networks

This section defines *state space model* (SSM) that is used for modeling dynamic systems of gene regulation. We consider gene expressions of p genes in time-course. Let $\mathcal{N} = \{0, 1, 2, \dots, T\}$ be the entire set of time points and let \mathcal{N}_{obs} be the set of time points where gene expressions are observed (measured). The level of gene expression of a gene g is represented as a real number and is called *gene expression value*. For example, when conventional DNA microarrays are used for observation, the gene expression value of g may represent the amount of mRNA expressions of g relative to some control. A *time-course gene expression data* of p genes is given as a series of p -dimensional vectors $y_n \in \mathcal{R}^p$ with $n \in \mathcal{N}_{\text{obs}}$, where \mathcal{R} is the set of real numbers. The set of time points where observations were not made is denoted by $\mathcal{N}_{\text{obs}}^c$. We call y_n an *observation variable*.

When modeling time-course gene expression data with linear Gaussian state space models, we assume a k -dimensional hidden *state variable* denoted by x_n for some appropriately chosen dimension k , and consider that the series of observation vectors $Y_{\mathcal{N}_{\text{obs}}} = \{y_n\}$ ($n \in \mathcal{N}_{\text{obs}}$) is generated in the following way:

$$\begin{aligned} x_n &= Fx_{n-1} + v_n \text{ for } n \in \mathcal{N}, \\ y_n &= Hx_n + w_n \text{ for } n \in \mathcal{N}_{\text{obs}}, \end{aligned}$$

where the first equation is called the *system model*, $F \in \mathcal{R}^{k \times k}$ is the *state transition matrix*, the second equation is called the *observation model*, $H \in \mathcal{R}^{p \times k}$ is the *observation matrix*, $v_n \sim N_k(0_k, Q)$ and $w_n \sim N_p(0_p, R)$ are the system noise and the observation noise, respectively. The initial state vector x_0 is assumed to be a Gaussian random vector with mean vector μ_0 and covariance matrix Σ_0 , i.e., $x_0 \sim N_k(\mu_0, \Sigma_0)$.

What we need to do is to estimate unknown parameters $\theta = \{H, F, Q, R, \mu_0\}$ for the system and observation models. However, the estimation of parameters

obviously would fail due to overfitting because the number of free parameters gets larger exponentially as the number p of genes increases. Conventionally, the length of time-course gene expression data is much shorter than the number of genes, i.e. $N = \mathcal{N}_{\text{obs}} \ll p$. Furthermore, the dimension k of state vector is also unknown and thus need to be determined to the optimal one. In [4,17], these difficulties were overcome to estimate gene networks underlying the observation gene expression data, where vector autoregressive models fail to estimate with the ordinary estimation procedure, e.g., the maximum likelihood estimation procedure. The parameters are not uniquely determined by the ordinal estimation procedure, since there are infinitely many parameter values which can yield the same likelihood. In order to avoid this, we need to reduce the degree of freedom of H , F and R . It is proven in [4] that if the following constraints are imposed on the parameters then the systems become identifiable: (i) $Q = I$. (ii) $H^T R^{-1} H = \Lambda \equiv \text{diag}(\lambda_1, \dots, \lambda_k)$ where $\lambda_1 > \dots > \lambda_k$. (iii) An arbitrary sign condition is imposed on the elements of the first row of H .

Then we utilize the EM algorithm with the constraints for the parameter estimation. In the algorithm, the conventional Kalman smoothing estimator of the state vectors obtained by Kalman filter and smoother algorithms are utilized. If x_n could represent the state of modules of genes, the estimated dimension k becomes smaller than the number of genes p . In this sense, this method can be seen as a *dimension reduction*. The SSM with the parameter θ is denoted by $\text{SSM}(\theta)$. Thus the parameter estimation problem is solved for $N \ll p$.

The next difficulty is how to represent the interactions between genes. This is solved as follows: By converting the estimated parameters and the model, a parsimonious representation of the first order vector autoregressive model is obtained as

$$R^{-1/2}(y_n - w_n) = \Psi R^{-1/2}(y_{n-1} - w_{n-1}) + R^{-1/2} H v_n,$$

where the autoregressive coefficient matrix is given by $\Psi \equiv D^T \Lambda F D$ with $D = \Lambda^{-1} H^T R^{-1/2}$. Since Ψ represents magnitude of interactions between genes, we can estimate a gene network with this equation. The technical details are referred to [4,17],

3 Gene Networks Viewed through State Space Model

This section shows two applications of SSM and their results in [4,16]. One is for inferring gene networks of human umbilical vein endothelial cells (HUVEC) undergoing growth factor deprivation-induced apoptosis. The other is an application of SSM to predict differences in gene regulatory systems of normal human small airway epithelial cells (SAEC) against an anti-cancer drug *Gefitinib*.

3.1 HUVEC Apoptosis Gene Network

Endothelial cell (EC) apoptosis (programmed suicide) may play an important role in blood vessel development, homeostasis and remodeling. In support of

this concept, EC apoptosis has been detected within remodeling vessels *in vivo*, and inactivation of EC apoptosis regulators has caused dramatic vascular phenotypes. The protein-based signaling and cleavage cascades that regulate EC apoptosis are well known. However, the possibility that programmed transcriptome and glycome changes contribute to EC apoptosis has only recently been explored. In order to understand the complex cause and effect relationships among these signals, we measured transcriptome of HUVECs with the time-course gene expression data [1] which were created using CodeLink 20k arrays and applied our SSM method to analyze this HUVEC apoptosis gene network.

mRNAs were prepared at 0.5, 1.5, 3, 6, 9, 12, 24h after the induction of apoptosis by growth factor deprivation. The experiments were repeated independently three times ($m = 3$). Among approximately 20,000 genes, we focused on 1048 genes. These genes are comprised of 48 genes known to play important roles in apoptosis and blood vessel development and 1000 genes giving the highest coefficients of variation. The real time set $\{0.5\text{h}, 1.5\text{h}, 3\text{h}, 6\text{h}, 9\text{h}, 12\text{h}, 24\text{h}\}$ is converted into $\mathcal{N}_{\text{obs}} = \{1, 3, 6, 12, 18, 24, 48\}$ and the entire set of time points is set to be $\mathcal{N} = \{1, 2, \dots, 47, 48\}$.

For this data, the BIC curves became monotone decreasing with respect to the dimension of state variable and we could not determine the dimension k . Instead, we took a heuristic approach based on the singular value decomposition of the projection matrix D (see [4] for the details) and reached a conclusion that it is reasonable to set $k = 4$. This is also supported by the discussions in [1] that suggested eight internal clusters of genes. Fig. 1 shows the estimated network focused on the most representative 50 genes, where the interactions around TRAF1 indicate biologically rational facts.

3.2 Systems Differences in SAEC Gefitinib Responses

If the parameters θ and observation data $Y_{\mathcal{N}_{\text{obs}}}$ are given, $\text{SSM}(\theta)$ can predict the observation y_n with the one-step-ahead prediction estimator

$$y_{n|n-1} = Hx_{n|n-1},$$

where $x_{n|n-1} = E(x_n | Y_{(n-1)})$ with $Y_{(n-1)} \subseteq Y_{\mathcal{N}_{\text{obs}}}$ which is the set of observations obtained before the n -th time step. Namely the prediction estimator predicts future observation with the previous observations in time course. The estimators are calculated sequentially by utilizing Kalman filter algorithm. To identify differentially regulated genes, we developed the following method to search genes which have unpredictable profiles in the case data by using a model for underlying dynamic system of the control data (see [16] for the technical details):

1. $\text{SSM}(\theta)$ is applied to the time-course gene expression data of the control $Y_{\mathcal{N}_{\text{obs}}}^{\text{CTRL}} = \{y_n^{\text{CTRL}}\}$, $n \in \mathcal{N}_{\text{obs}}$ and the parameters are estimated. As a result, a model for dynamic system of the control data, $\text{SSM}(\hat{\theta}^{\text{CTRL}})$, is obtained.
2. $\text{SSM}(\hat{\theta}^{\text{CTRL}})$ is applied to predict time-course gene expression data of the case $Y_{\mathcal{N}_{\text{obs}}}^{\text{CASE}} = \{y_n^{\text{CASE}}\}$, $n \in \mathcal{N}_{\text{obs}}$.

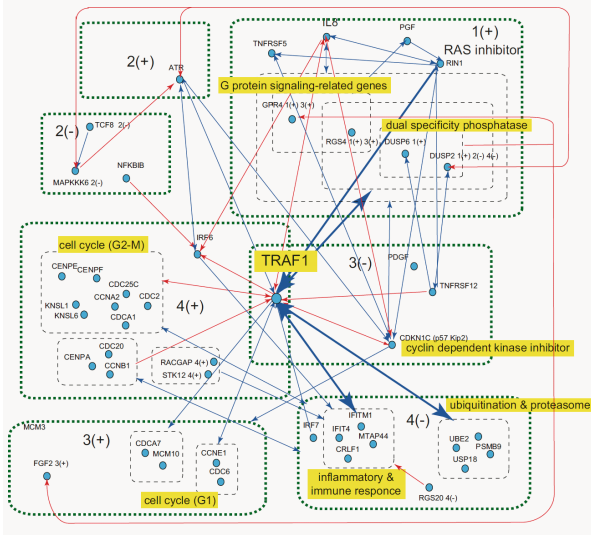


Fig. 1. HUVEC apoptosis gene network around TRAF1. \mathcal{M}_{i+} and \mathcal{M}_{i-} tend to exhibit the opposite expression patterns for $i = 1, \dots, 4$.

3. In order to identify differentially regulated genes, we search genes whose expressions are not well predicted for the case data but well predicted for the control data by using the control model $\text{SSM}(\hat{\theta}^{\text{CTRL}})$.

We measured time-course gene expression data of SAECs by using Agilent Whole Human Genome Oligo Microarray (G4112F) and used Gefitinib (GFT) as a drug to see its responses. GFT is known as a selective inhibitor of epidermal growth factor receptor’s (EGFR) signaling pathway. We prepared the following samples:

- SAEC treated with epidermal growth factor (EGF), which is used as control and labeled with “EGF”.
- SAEC treated with both EGF and GFT, which is used as case and labeled with “EGF-GFT”.

For each sample, after starvation to synchronize the cell cycle, we took 19 time points during 48 hours at time 0, 0.5, 1, 2, 3, 4, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 43, 48 [hour]. Thus we set $\mathcal{N}_{\text{obs}} = \{1, 2, 3, 5, 9, 13, 19, 25, 31, 37, 43, 49, 55, 61, 67, 73, 79, 87, 97\}$ and $\mathcal{N} = \{1, \dots, 97\}$. GFT was dosed two hours before the 0 hour to the case sample. EGF was dosed at the 0 hour to the both samples. We denote the time-course gene expression data for “EGF” and “EGF-GFT” by $Y_{\mathcal{N}_{\text{obs}}}^{\text{EGF}}$ and $Y_{\mathcal{N}_{\text{obs}}}^{\text{EGF-GFT}}$, respectively. The parameters $\theta = \{H, F, R, \mu_0\}$ were estimated for $Y_{\mathcal{N}_{\text{obs}}}^{\text{EGF}}$ and $Y_{\mathcal{N}_{\text{obs}}}^{\text{EGF-GFT}}$. During the course of estimation, the dimension of state vector was determined to $k = 9$ and we obtained the dynamic models represented as SSMs for the control data (EGF) and the case data (EGF-GFT): $\text{SSM}(\hat{\theta}^{\text{EGF}})$ and $\text{SSM}(\hat{\theta}^{\text{EGF-GFT}})$.

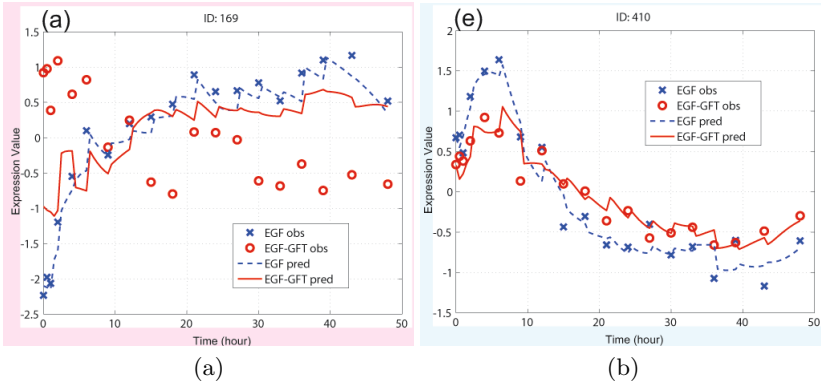


Fig. 2. The time-course profiles of genes: (a) g_{169} , (b) g_{410} . The observed data points for EGF (EGF-GFT) are represented by \times (\circ). The predicted observations for EGF (EGF-GFT) by $SSM(\theta^{EGF})$ are represented by dashed (solid) lines.

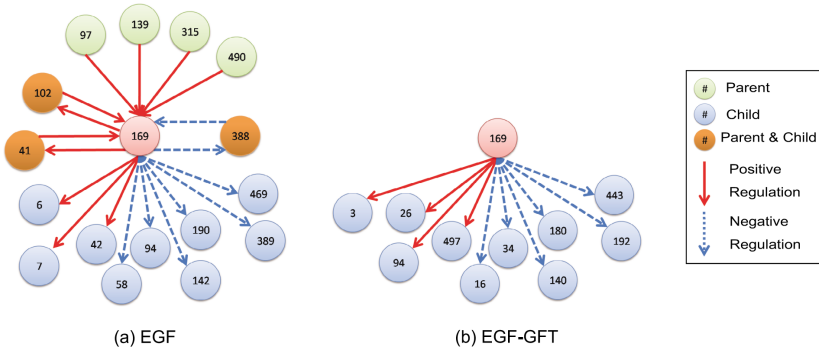


Fig. 3. Resulting gene networks around a significantly differentially regulated gene g_{169} (see Fig. 2(a)) estimated from (a) EGF data, and (b) EGF-GFT data using SSMs. The edges are with the weights $|\psi_{ij}| > 0.015$.

With this prediction strategy, we utilized predictability and unpredictability for gene expression profiles generated from a gene regulatory system by using a dynamic gene regulatory model for another system. We identified such unpredictable genes as candidates for differentially regulated genes from the data sets of SAEC treated with EGF and GFT. Gene g_{169} is a typical example which is significantly differentially regulated while g_{410} is insignificantly differentially regulated (Fig. 2). Fig. 3 shows gene regulatory networks around the identified gene g_{169} .

4 Dynamic Bayesian Networks with Nonparametric Regression

Bayesian network is a mathematical model for representing causal relationships among random variables by using conditional probabilities. Bayesian network

assumes a directed acyclic graph (DAG), denoted by G , as a relationship among random variables, and mostly discrete random variables have been used for modeling causal relationships [3].

4.1 General Framework

We first define a general framework of *Bayesian network with nonparametric regression* for modeling gene networks from gene expression data based on perturbations such as gene knockdowns and drug doses, where a gene is regarded as a random variable taking a continuous number and a causal relationship means regulation [5,6]. In this general framework, gene expression data need not be in time-course.

We consider p genes for gene networks. Then let X_i ($i = 1, \dots, p$) be a random variable that takes a value from \mathcal{R} . If there is a directed edge e_{ij} from X_i to X_j , we say that X_i is a parent of X_j . We denote by $Pa(X_i) \subset \{X_1, \dots, X_p\}$ the set of parents of X_i in G . In the DAG G , the random variable X_i only depends on its direct parents in $Pa(X_i)$ and is independent of other variables, i.e., this offers the first order *Markov property* to the relationship among variables described by G . Using the DAG G and its Markov property, the joint probability of all random variables can be decomposed as the product of conditional probabilities:

$$P(X_1, \dots, X_p) = \prod_{j=1}^p P(X_j | Pa(X_j)). \quad (1)$$

The conditional probabilities $P(X_j | Pa(X_j))$ describe the parent-child relationships. If we know the true structure of G *a priori*, from Eq. 1, we can construct the joint probability function by estimating each conditional probability. However, in the gene network estimation, the true G is not known and this problem can be considered as a statistical model selection problem.

Suppose that we have a data set $\mathbf{X}_n = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ of n values of p -dimensional random variable vector $\mathbf{X} = (X_1, \dots, X_p)^t$, where $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^t$ corresponds to the vector of p gene expression values by the i -th observation, e.g., the i -th microarray measurement. Here \mathbf{a}^t represents the transpose of \mathbf{a} . Using the data \mathbf{X}_n , we can rewrite Eq. 1 using densities instead of the probabilistic measure:

$$f(\mathbf{x}_1, \dots, \mathbf{x}_n | \boldsymbol{\theta}, G) = \prod_{i=1}^n \prod_{j=1}^p f_j(x_{ij} | \mathbf{p}_{ij}, \boldsymbol{\theta}_j), \quad (2)$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^t, \dots, \boldsymbol{\theta}_p^t)^t$ is the parameter vector and \mathbf{p}_{ij} is the gene expression value vector of the parents of X_j by the i -th observation. The construction of the conditional probability $f_j(x_{ij} | \mathbf{p}_{ij}, \boldsymbol{\theta}_j)$ is equivalent to the problem of the fitting regression model to the data $\{(x_{ij}, \mathbf{p}_{ij}) \mid i = 1, \dots, n\}$ by $x_{ij} = m_j(\mathbf{p}_{ij}) + \varepsilon_{ij}$, where $m_j(\cdot)$ is a smooth function from $\mathcal{R}^{|Pa(X_j)|}$ to \mathcal{R} and ε_{ij} ($i = 1, \dots, n$) are independently and normally distributed with mean 0 and variance σ_j^2 . $|Pa(X_j)|$

denotes the number of elements in $Pa(X_j)$. If we set the function $m_j(\cdot)$ by $m_j(\mathbf{p}_{ij}) = \beta_0 + \boldsymbol{\beta}^t \mathbf{p}_{ij}$, where β_0 and $\boldsymbol{\beta} = (\beta_1, \dots, \beta_{|Pa(X_j)|})^t$ are parameters, we have a linear regression model to capture the relationship between x_{ij} and \mathbf{p}_{ij} in Friedman et al. [3]. To capture even nonlinear dependencies, Imoto et al. [5,6] proposed the use of the *nonparametric additive regression model* of the following form:

$$x_{ij} = m_{j,1}(p_{i,1}^{(j)}) + \dots + m_{j,|Pa(X_j)|}(p_{i,|Pa(X_j)|}^{(j)}) + \varepsilon_{ij} \quad (3)$$

where $m_{j,k}(\cdot)$ ($k = 1, \dots, |Pa(X_j)|$) are smooth functions from \mathcal{R} to \mathcal{R} and $\mathbf{p}_{ij} = (p_{i,1}^{(j)}, \dots, p_{i,|Pa(X_j)|}^{(j)})^t$. We construct $m_{j,k}(\cdot)$ by the basis function expansion method with *B-splines*: $m_{j,k}(p) = \sum_{s=1}^{M_{jk}} \gamma_{sk}^{(j)} b_{sk}^{(j)}(p)$, where $\gamma_{sk}^{(j)}$ are parameters, $\{b_{1k}^{(j)}(\cdot), \dots, b_{M_{jk}k}^{(j)}(\cdot)\}$ is the prescribed set of *B-splines*, and M_{jk} is the number of *B-splines*.

By combining Eqs. [2] and [3], we have a Bayesian network model with *B-spline nonparametric regression* of the form

$$f(\mathbf{X}_n | \boldsymbol{\theta}, G) = \prod_{i=1}^n \prod_{j=1}^p \frac{1}{(2\pi\sigma_j^2)^{1/2}} \exp \left\{ -\frac{(x_{ij} - \sum_k \sum_s \gamma_{sk}^{(j)} b_{sk}^{(j)}(p_{ik}^{(j)}))^2}{2\sigma_j^2} \right\}. \quad (4)$$

Once a graph structure G is given, the statistical model based on Eq. [4] can be estimated by a suitable procedure. In order to find the best graph structure, we define a criterion for evaluating a graph based on our model from Bayes approach, that is the maximization of the *posterior probability* of the graph $P(G | \mathbf{X}_n)$. The posterior probability of a graph is written by $P(G | \mathbf{X}_n) = p(\mathbf{X}_n | G)P(G)/p(\mathbf{X}_n) \propto p(\mathbf{X}_n | G)P(G)$, where $P(G)$ is the *prior probability* of the graph and $p(\mathbf{X}_n)$ is the *normalizing constant* and not related to the graph selection. The likelihood $p(\mathbf{X}_n | G)$ is obtained by marginalizing the joint density $p(\mathbf{X}_n, \boldsymbol{\theta} | G)$ against $\boldsymbol{\theta}$ and given by

$$p(\mathbf{X}_n | G) = \int f(\mathbf{X}_n, \boldsymbol{\theta} | G) d\boldsymbol{\theta} = \int f(\mathbf{X}_n, \boldsymbol{\theta} | G) p(\boldsymbol{\theta} | \boldsymbol{\lambda}, G) d\boldsymbol{\theta}, \quad (5)$$

where $p(\boldsymbol{\theta} | \boldsymbol{\lambda}, G)$ is the prior distribution on the parameter $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ is the hyperparameter vector. Under the Bayes approach, we can choose the optimal graph such that $p(\mathbf{X}_n | G)$ is maximum. A crucial problem for constructing a criterion based on the posterior probability of the graph is the computation of the high-dimensional integration in Eq. [5]. For $\log p(\boldsymbol{\theta} | \boldsymbol{\lambda}, G) = O(n)$, the Laplace approximation for integrals gives an analytical solution:

$$\begin{aligned} & \int f(\mathbf{X}_n | \boldsymbol{\theta}, G) p(\boldsymbol{\theta} | \boldsymbol{\lambda}, G) d\boldsymbol{\theta} \\ &= \frac{(2\pi/n)^{r/2}}{|J_{\boldsymbol{\lambda}}(\hat{\boldsymbol{\theta}} | \mathbf{X}_n)|^{1/2}} \exp\{nl_{\boldsymbol{\lambda}}(\hat{\boldsymbol{\theta}} | \mathbf{X}_n)\} \{1 + O_p(n^{-1})\}, \end{aligned} \quad (6)$$

where r is the dimension of $\boldsymbol{\theta}$, $l_{\boldsymbol{\lambda}}(\hat{\boldsymbol{\theta}} | \mathbf{X}_n) = \{\log f(\mathbf{X}_n | \boldsymbol{\theta}, G) + \log p(\boldsymbol{\theta} | \boldsymbol{\lambda}, G)\}/n$, $J_{\boldsymbol{\lambda}}(\boldsymbol{\theta} | \mathbf{X}_n) = -\partial^2 l_{\boldsymbol{\lambda}}(\boldsymbol{\theta} | \mathbf{X}_n) / \partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^t$, and $\hat{\boldsymbol{\theta}}$ is the mode of $l_{\boldsymbol{\lambda}}(\boldsymbol{\theta} | \mathbf{X}_n)$. Hence, by

taking minus twice logarithm of $P(G|\mathbf{X}_n)$ and substituting Eqs. 5 and 7 into $P(G|\mathbf{X}_n)$, Imoto et al. [5] derived a criterion called BNRC (*Bayesian network and Nonparametric Regression Criterion*) for choosing the optimal graph:

$$\text{BNRC}(G) = -2 \log P(G) - r \log(2\pi/n) + \log |J_\lambda(\hat{\theta}|\mathbf{X}_n)| - 2nl_\lambda(\hat{\theta}|\mathbf{X}_n). \quad (7)$$

The optimal graph \hat{G} is chosen so that the criterion Eq. 7 is minimal. Imoto et al. [6] also extended the results in [5] to handle the *nonparametric heteroscedastic regression*. In practice, the value of $\text{BNRC}(G)$ defined in Eq. 7 can be computed by the sum of the local scores, $\text{BNRC}(G) = \sum_{j=1}^p \text{BNRC}_j$, where BNRC_j is defined by the approximation of

$$-2 \log P_j(G) \int \prod_{i=1}^n f_j(x_{ij}|\mathbf{p}_{ij}, \theta_j) p_j(\theta_j|\lambda_j) d\theta_j$$

obtained by the Laplace approximation. Here, we assume $p(\theta|\lambda, G) = \prod_{j=1}^p p_j(\theta_j|\lambda_j)$ and $P_j(G)$ is called the prior probability for the j -th local structure defined by the j -th random variable and its direct parents. Note that $P(G) = \prod_{j=1}^p P_j(G)$ holds.

There is obviously a limit from a biological viewpoint on the gene network construction only from gene expression data (mRNA expression). To overcome this limit, Imoto et al. [7] also developed a general framework for combining gene expression data and other biological data such as protein-protein and protein-DNA interactions, sequences of the binding site of the genes controlled by transcription regulators, literature and so on.

Finding optimal Bayesian networks is computationally hard. It is known that determining the optimal network is NP-hard [2]. However, for a small number of genes such as 30, it is possible to search for optimal graph structures on a computer with 200 GFLOPS. Ott et al. [11][12] devised an algorithm of time complexity $O(p2^p)$ which runs in practice on such computers, where p is the number of nodes. For searching large networks, we need to employ greedy hill-climbing algorithms to learn graph structure such as in [3][5]. Recently, Perrier et al. [13] devised an algorithm which finds optimal networks with the constraint on the number of candidate parents in time $O(p2^M)$, where $M = \max_{j=1}^p \{m_j \mid m_j \text{ is the number of candidate parents}\}$. This allows us to compute optimal gene networks of size 80 on a computer with 200 GFLOPS in practice.

4.2 Dynamic Bayesian Networks

A shortcoming of Bayesian network is that its network structure is acyclic, while a real gene regulation mechanism has cyclic regulations. The use of the *dynamic Bayesian networks* (DBN) is yet useful for constructing a gene network with cyclic regulations although SSM has already resolved this difficulty. In the context of modeling gene networks by dynamic Bayesian network, we consider a time-course gene expression data, i.e., the t -th gene expression data \mathbf{x}_t corresponds to the state of p genes at time t . \mathbf{x}_t is considered as an observation of the

p -dimensional random vector \mathbf{X}_t . As for the time dependency, we consider the first order Markov relation described as $\mathbf{X}_1 \rightarrow \mathbf{X}_2 \rightarrow \cdots \mathbf{X}_n$. Then the joint probability is decomposed as follows:

$$P(\mathbf{X}_1, \dots, \mathbf{X}_T) = P(\mathbf{X}_1)P(\mathbf{X}_2|\mathbf{X}_1) \cdots P(\mathbf{X}_T|\mathbf{X}_{T-1}). \quad (8)$$

The gene regulations can be modeled through the construction of $P(\mathbf{X}_t|\mathbf{X}_{t-1})$ for $t = 2, \dots, T$. In this definition of dynamic Bayesian network, it is assumed that the structure of gene networks are stable through all time points. This notion of DBN will be used in an extensive manner in Section 5 to capture the dynamically changing networks structures in time-course. The conditional probability can also be decomposed into the product of conditional probabilities of each gene given its parents

$$P(\mathbf{X}_t|\mathbf{X}_{t-1}) = \prod_{j=1}^p P(X_{tj}|Pa(X_j)_{t-1}), \quad (9)$$

where $Pa(X_j)_{t-1}$ is the set of random variables corresponding to the parent genes of the j -th gene at time $t - 1$. By combining Eqs. 8 and 9, we have the decomposition:

$$P(\mathbf{X}_1, \dots, \mathbf{X}_T) = P(\mathbf{X}_1) \prod_{t=2}^T \prod_{j=1}^p P(X_{tj}|Pa(X_j)_{t-1}). \quad (10)$$

We can define the BNRC score named $\text{BNRC}_{\text{dynamic}}$ for dynamic Bayesian network with nonparametric regression in a similar way. This extension of dynamic Bayesian networks combined with nonparametric regression allows us to detect nonlinear relationships while the modeling by SSM in Section 2 can capture only linear features. This is an advantage of DBN with nonparametric regression. The construction of a graph selection criterion based on the Bayes approach can be done in the same way as the Bayesian networks in the previous section. The details of the combination of the dynamic Bayesian networks with the nonparametric regression are described in Kim et al. [9].

5 Gene Networks Viewed through Dynamic Bayesian Network with Nonparametric Regression

This section shows an application of dynamic Bayesian network to analyze the activities of autocrine pathways in HUVEC controlling transcriptome network against a drug called Fenofibrate [15].

5.1 Super Dynamic Bayesian Network

For a time-course gene expression data based on drug response, it is often observed that some subnetwork at some time points shows its high activities and

transmit signals to another subnetwork at some later time points. In order to model such dynamics of drug response transcriptome networks, we extended the dynamic Bayesian network with nonparametric regression called *super dynamic Bayesian network* (SDBN) so that it can capture this feature [15]. The idea of SDBN is to define the *active gene set* for each time point and to define the node set at each time point for network modeling. We call this method *node-set separation method*. We say that a gene g_i is *active* at time t if it is expressed differentially comparing with the controls, namely, $\text{pv}(g_i, t) \leq \theta_t$, where $\text{pv}(g_i, t)$ is the p-value of g_i at time t , and θ_t is the threshold for time t that could be determined by using false discovery rate, for example. In our paper [15], the p-value of g_i at time t is computed by comparing triplet expression values of the gene g_i at time t with control four replicate expression values, i.e., expression data of cells which are not treated with the drug. We denote by \mathcal{A}_t the set of active genes at time t . Then we define the *node set* for time t by $\mathcal{N}_t = \mathcal{A}_{t-1} \cup \mathcal{A}_t$ for $t = 1, \dots, T$, where \mathcal{A}_0 is the empty set.

The definition of the node set has the basis on the Markov process of the dynamic Bayesian networks, i.e., the DBN assumes the first order Markov process among time-course data as Eq. 8. The *transcriptome network* at time t , denoted by G_t , is estimated for the node set \mathcal{N}_t by the DBN and nonparametric regression with whole expression data at time points $t = 1, \dots, T$. Finally the dynamic transcriptome network is obtained by $G = G_1 \cup \dots \cup G_T$. The advantage of this estimation procedure, i.e., using node set \mathcal{N}_t separately instead of using all nodes $\mathcal{N} = \mathcal{A}_1 \cup \dots \cup \mathcal{A}_T$ as the node set, is not only finding dynamics of transcriptome networks, but also possibility to reduce false positive edges in the network, because we can reduce the size of the gene set for each observed time efficiently and this will increase the accuracy of the structure learning.

5.2 Dynamic Transcriptome Network

This method was applied to search autocrine pathways from time-course drug response gene expression data and protein-protein interaction (PPI) data [15]. Fenofibrate is an anti-hyperlipidemia drug and it is an agonist of the peroxisome proliferator-activated receptor α (PPAR α), which is known as a transcription factor that induces genes related to the lipid metabolism. Recent studies revealed that Fenofibrate has anti-inflammatory effects [14]. We aimed at identifying Fenofibrate-affected autocrine pathways related to its anti-inflammatory effects, and at elucidation of unknown modes-of-action.

We used CodeLink Human UniSet I 20K arrays for measuring drug-response time course and 400 siRNA gene knockdown expression data of HUVEC. For the time-course data, we observed 6 time points including the control, 2, 4, 6, 8 and 18 hours after treated with 25 μM Fenofibrate in 3 or 4 replicates. We excluded probes which have less than 90% G flags for all 400 arrays from the knockdown expression data. Missing values which do not have G flags were imputed by LSImput (<http://www.ii.uib.no/~trondb/imputation/>). For PPI data, we used the data set publicly available in Genome Network Platform (GNP) (released on May 27 2008: <http://genomenetwork.nig.ac.jp/>). By removing proteins which

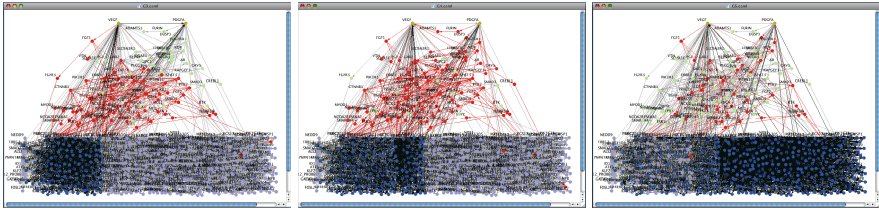


Fig. 4. Extracted autocrine ligand pathways for G_3 (left), G_4 (center), and G_5 (right). The top two nodes VEGF and PDGFA are ligand genes in the network. Nodes and edges in the bottom are the transcriptome network. The middle part contains the proteins and the extracted significant PPI pathways. The large version of this figure and complete data are available on the online supplement (<http://bonsai.ims.u-tokyo.ac.jp/~tamada/suppl/PSB2009/>). Transcriptome network edges are also included in the middle part since the two ligands are also involved in the transcriptome networks.

do not have the corresponding probes in the microarray, the final PPI network contains 42,570 edges for 9,016 proteins. This PPI network contains 308 receptor and 149 ligand (in total 457) proteins.

We selected genes whose SAM q value $\leq 5\%$ and fold change ≥ 1.5 at time t for \mathcal{A}_t where $t = 1, \dots, 5$ corresponding to 2hr, 4hr, 6hr, 8hr and 18hr, respectively. See <http://www-stat.stanford.edu/~tibs/SAM/> for SAM. If a gene has more than one probe in the microarrays, we selected the one that has the smallest average of SAM p values for all the time points. This is required for the steps to match each probe in the microarrays to a protein in the PPI network. Finally, the numbers of genes in \mathcal{A}_t are 14, 5, 144, 129 and 370, respectively. The numbers of genes in \mathcal{N}_t are 14, 19, 144, 200, and 454, respectively. The total number of unique genes in the network is 527.

The transcriptome networks G_t ($t = 1, \dots, 5$) were estimated with the prior networks which were estimated from knockdown gene expression data to incorporate transcriptome level changes which can be observed by knockdown genes by siRNAs [11]. The reliability of the edges in the estimated networks is calculated by the bootstrap method [8] with 1000 iterations. Edges whose bootstrap probability is less than threshold 0.05 were removed from the final transcriptome networks. Fig. 4 shows dynamic changes of the networks from which we extracted 23 autocrine-like pathways including VEGF–NRP1–GIPC1–PRKCA–PPAR α , that is one of the most significant ones and contains PPAR α , a target of Fenofibrate. The details of the computational method are referred to [15].

References

1. Affara, M., Dunmore, B., Savoie, C.J., Imoto, S., Tamada, Y., Araki, H., Charnock-Jones, D.S., Miyano, S., Print, C.: Understanding endothelial cell apoptosis: What can the transcriptome glycome and proteome reveal? *Philosophical Transactions of Royal Society B* 362(1484), 1469–1487 (2007)

2. Chickering, D.M.: Learning Bayesian networks is NP-complete. In: Fisher, D., Lenz, H.-J. (eds.) *Learning from Data: Artificial Intelligence and Statistics V*, pp. 121–130. Springer, Heidelberg (1996)
3. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyze expression data. *J. Comput. Biol.* 7(3-4), 601–620 (2000)
4. Hirose, O., Yoshida, R., Imoto, S., Yamaguchi, R., Higuchi, T., Charnock-Jones, D.S., Print, C., Miyano, S.: Statistical inference of transcriptional module-based gene networks from time course gene expression profiles by using state space models. *Bioinformatics* 24(7), 932–942 (2008)
5. Imoto, S., Goto, T., Miyano, S.: Estimation of genetic networks and functional structures between genes by using Bayesian network and nonparametric regression. In: *Pacific Symposium on Biocomputing*, vol. 7, pp. 175–186 (2002)
6. Imoto, S., Kim, S., Goto, T., Aburatani, S., Tashiro, K., Kuhara, S., Miyano, S.: Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. *J. Bioinf. Comp. Biol.* 1(2), 231–252 (2003)
7. Imoto, S., Higuchi, T., Goto, T., Tashiro, K., Kuhara, S., Miyano, S.: Combining microarrays and biological knowledge for estimating gene networks via Bayesian networks. *J. Bioinf. Comp. Biol.* 2(1), 77–98 (2004)
8. Imoto, S., Tamada, Y., Araki, H., Yasuda, K., Print, C.G., Charnock-Jones, D.S., Sanders, D., Savoie, C.J., Tashiro, K., Kuhara, S., Miyano, S.: Computational strategy for discovering druggable gene networks from genome-wide RNA expression profiles. In: *Pacific Symposium on Biocomputing*, vol. 11, pp. 559–571 (2006)
9. 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(1-3), 57–65 (2004)
10. Kitagawa, G., Gersch, W.: *Smoothness priors analysis of time series*. Springer, New York (1996)
11. Ott, S., Imoto, S., Miyano, S.: Finding optimal models for small gene networks. In: *Pacific Symp. Biocomput.*, vol. 9, pp. 557–567 (2004)
12. Ott, S., Hansen, A., Kim, S.-Y., Miyano, S.: Superiority of network motifs over optimal networks and an application to the revelation of gene network evolution. *Bioinformatics* 21(2), 227–238 (2005)
13. Perrier, E., Imoto, S., Miyano, S.: Finding optimal Bayesian network given a superstructure. *J. Machine Learning Research* 9, 2251–2286 (2008)
14. Straus, D.S., Glass, C.K.: Anti-inflammatory actions of PPAR ligands: new insights on cellular and molecular mechanisms. *Trends Immunol* 28(12), 551–558 (2007)
15. Tamada, Y., Araki, H., Imoto, S., Nagasaki, M., Doi, A., Nakanishi, Y., Tomiyasu, Y., Yasuda, K., Dunmore, B., Sanders, D., Humphreys, S., Print, C., Charnock-Jones, D.S., Tashiro, K., Kuhara, S., Miyano, S.: Unraveling dynamic activities of autocrine pathways that control drug-response transcriptome networks. In: *Pacific Symposium on Biocomputing*, vol. 14, pp. 251–263 (2009)
16. Yamaguchi, R., Imoto, S., Yamauchi, M., Nagasaki, M., Yoshida, R., Shimamura, T., Hatanaka, Y., Ueno, K., Higuchi, T., Gotoh, N., Miyano, S.: Predicting differences in gene regulatory systems by state space models. *Genome Informatics* 21, 101–113 (2008)
17. Yamaguchi, R., Yoshida, R., Imoto, S., Higuchi, T., Miyano, S.: Finding module-based gene networks with state-space models – Mining high-dimensional and short time-course gene expression data. *IEEE Signal Processing Magazine* 24(1), 37–46 (2007)

Identifying Evolutionarily Conserved Protein Interaction Modules Using GraphHopper

Corban G. Rivera* and T.M. Murali**

114 McBryde Hall, Department of Computer Science,
Virginia Polytechnic Institute and State University, Blacksburg VA 24061
cgrivera@vt.edu, murali@cs.vt.edu

Abstract. We study the question of detecting Conserved Protein Interaction Modules (CPIMs) in protein-protein interaction (PPI) networks. We propose a novel algorithm called GraphHopper that analyzes two PPI networks to find CPIMs. GraphHopper finds CPIMs by “hopping” from one network to another using orthology relationships. By decoupling the degree of evolutionary conservation in a CPIM from the reliability of the PPIs in a CPIM, GraphHopper finds CPIMs with a wide variety of topologies that previous algorithms cannot detect.

GraphHopper is competitive with NetworkBlast and Match-and-Split, two state-of-the-art algorithms for computing CPIMs, on the task of recapitulating MIPS processes and complexes. Upon applying GraphHopper to human, fly, and yeast PPI networks, we find a number of CPIMs involved in fundamental processes of the cell that are conserved in all three species. We present the first global map of human-fly CPIMs. This map sheds light on the conservation of protein interaction modules in multi-cellular organisms. CPIMs related to development and the nervous system emerge only in the human-fly comparison. For example, a set of 10 interconnected CPIMs suggest that fly proteins involved in eye development may have human orthologs that have evolved functions related to blood clotting, vascular development, and structural support.

1 Introduction

Protein-Protein Interaction (PPI) networks containing thousands of interactions are now available for a number of species, including human, yeast, worm, and fly. Pairwise comparison of these networks enables the computation of groups of interacting proteins that are conserved in different organisms [1], thus laying the basis for module-level modeling of cellular processes. Such conserved sets consist of two connected protein interaction sub-networks (or modules), one in each PPI network, such that proteins in each module have orthologs in the other module. In this paper, we call these *Conserved Protein Interaction Modules* (CPIMs).

* Current address: Johns Hopkins University, School of Medicine, 720 Rutland Ave, Traylor 613, Baltimore, MD 21205-2109.

** Corresponding author.

Sharan and Ideker [1] survey many techniques developed to address this problem. A common feature of a number of these approaches [2,3,4] is the combination of the PPI networks of two species into a single “alignment graph”. A node in the alignment graph represents two orthologous proteins, one from each PPI network. An edge in the alignment graph represents an interaction that is conserved in both PPI networks. These methods add an edge to the alignment graph only if the proteins contributing to the nodes are connected through at most one intermediate protein in the respective PPI networks. The weight of an edge represents the likelihood that the corresponding interactions are conserved; this weight depends on the degree of orthology between the proteins and on assessed confidence estimates that the individual PPIs indeed take place in the cell. These authors find CPIMs by using various approaches to compute paths, complexes, and subgraphs of high weight in the alignment network and then expanding each such subgraph into the constituent PPIs.

Our approach. In this paper, we present a novel algorithm called GraphHopper for computing CPIMs in two PPI networks. GraphHopper computes two scores of quality for each CPIM. (i) The *conservation score* measures the total amount of sequence similarity among the proteins in the CPIM, averaged over the number of proteins in the CPIM. (ii) The *unreliability score* measures our total confidence in all the PPIs in the CPIM; this measure is useful since it is well documented that a number of high-throughput assays for detecting PPIs have high error rates [5]. A “good” CPIM has high conservation score and low unreliability score. Unlike the techniques mentioned earlier, GraphHopper treats the two PPI networks separately and connects each node in one PPI network to its potential orthologs in the other PPI network. GraphHopper starts by constructing a number of basis CPIMs, each of which is a pair of orthologous protein-pairs that directly interact. GraphHopper then expands each basis CPIM into a CPIM by “hopping” from one PPI network to another. In each hop, GraphHopper adds proteins and interactions to the current CPIM while ensuring that (i) the conservation score does not decrease and (ii) the unreliability score increases as little as possible. GraphHopper stops when it cannot add any more proteins without decreasing the conservation score.

Like GraphHopper, Narayanan and Karp’s Match-and-Split algorithm [6] does not construct an alignment network. They use combinatorial criteria to decide when the local neighborhoods of a pair of orthologs match. Under their model, they prove that a given pair of proteins can belong to at most one CPIM. This observation leads to a top-down partitioning algorithm that finds all maximal CPIMs in polynomial time. The MULE algorithm developed by Koyuturk et al. [7] also keeps PPI networks separate; it uses ortholog contraction and frequent subgraph detection to identify CPIMs.

Our contributions. We used GraphHopper to analyze all pairwise combinations of human, fly, and yeast PPI networks. Other approaches have considered the conservation of human PPIs and CPIMs in the networks of other eukaryotes [3,7,8]. The primary contribution of our work is a significant expansion of

these results by (i) considering a dataset of human PPIs integrated from multiple sources, (ii) detecting large functionally-enriched CPIMs with diverse topologies, and (iii) computing an integrated high-level map of CPIMs conserved only in human and fly PPI networks. As far as we know, this paper is the first to construct such a high-level map of human-fly CPIMs. Modules found by Gandhi et al. [8] were restricted to fundamental processes of life such as DNA replication and repair and transcription. In contrast, we find many CPIMs with that are enriched in functions unique to multi-cellular organisms. For instance, we find a set of 10 interconnected CPIMs which suggest that fly proteins involved in eye development may have human orthologs that have evolved functions related to blood clotting, vascular development, and structural support.

We compared GraphHopper to NetworkBlast [9], a state-of-the-art algorithm based on alignment networks, and to Match-and-Split [6]. We measured the ability of these three algorithms to recover MIPS complexes and processes [10]. In general, GraphHopper is competitive with and sometimes outperforms Match-and-Split in spite of computing a much larger number of CPIMs. GraphHopper has better precision and recall than NetworkBlast for MIPS processes.

An important feature of GraphHopper is its ability to compute CPIMs of far more diverse topologies than algorithms based on alignment networks. Algorithms that operate on alignment networks compute (highly weighted) subgraphs and map them into modules in each PPI network being compared. Since each such module is likely to have a topology very similar to the subgraph in the alignment network, the modules themselves have very similar topologies. In contrast, GraphHopper keeps the two PPI networks separate, thereby decoupling the evolutionary conservation of the proteins in a CPIM from the reliability of the PPIs that connect the proteins. As a result, GraphHopper is able to adapt to differing patterns of interactions in the two PPI networks, e.g., matching a module with one topology (say, a star) in one PPI network to a module with a considerably different topology (say, a complex) in the other PPI network. GraphHopper outperforms not just NetworkBlast but also Match-and-Split in this comparison. Match-and-Split only outputs CPIMs with at most 50 proteins; this restriction may hamper its ability to find CPIMs with diverse topologies.

2 Algorithms

2.1 A Model for CPIMs

We represent the set of PPIs in an organism as an undirected graph $G(V, E)$, where V is the set of proteins in the organism and each edge $(a, b) \in E$ is an interaction between proteins a and b . We associate a weight l_e with each edge $e \in E$. Let $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ be the PPI networks of two different organisms. We represent orthologous proteins as a bipartite graph H in which each edge $(a, b) \in V_1 \times V_2$ represents a pair of orthologous proteins a and b . Each edge $e \in H$ has a weight w_e equal to the BLASTP E -value between the connected proteins. We define a *Conserved Protein Interaction Module* (CPIM) as a triple (T_1, T_2, O) where T_1 and T_2 are connected subgraphs of G_1 and G_2 ,

respectively, and $O \subseteq H$ such that $(a, b) \in O$ if and only if a is a node in T_1 and b is a node in T_2 . Thus, O is the subgraph of H induced by the nodes in T_1 and T_2 . We define two quantities to measure the quality of a CPIM.

Conservation score. The conservation score of a CPIM (T_1, T_2, O) measures the amount of evolutionary similarity (at the amino acid level) between the protein interaction networks T_1 and T_2 . Let P_1 (respectively, P_2) be the set of nodes in T_1 (respectively, T_2). We define the *conservation score* of a CPIM (T_1, T_2, O) as

$$\phi(T_1, T_2, O) = \frac{\sum_{e \in O} -\log(w_e)}{|P_1| + |P_2|}.$$

The larger this score, the more evolutionary conserved T_1 and T_2 are since fewer proteins without orthologs can belong to the CPIM.

Unreliability score. Since many experimental techniques used to detect PPIs are error-prone, a number of methods have been developed to assess PPI reliabilities [5]. We do not consider methods that use gene expression data, since our goal is detect conservation purely at the level of PPIs. We also discard techniques that use functional annotations, since we use this data to assess the biological information in a CPIM. Therefore, we compute edge weights using the method proposed by Goldberg and Roth [11]: if the two nodes incident on an edge have more common neighbors than would be expected by chance, they assigned a high confidence (low p -value) to that edge. For a PPI e , we compute this p -value p_e using Fischer’s exact test and set $l_e = -\log(1 - p_e)$. We use Bonferroni’s correction to adjust for multiple hypotheses testing. We define the *unreliability score* $q(T_1, T_2, O)$ of a CPIM as follows:

$$q(T_1, T_2, O) = \sum_{e \in T_1 \cup T_2} l_e.$$

Since p_e is a probability, we combine the weights of multiple PPIs by adding their logarithms (the l_e values). A CPIM with high confidence edges has a small unreliability score.

2.2 The GraphHopper Algorithm

The GraphHopper algorithm finds CPIMs with high conservation and low unreliability scores. Our inputs are two protein interaction networks $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ and a set of orthologous protein pairs H . We define the *lightness* of a path π of PPIs in G_1 or G_2 to be $|\pi| = \sum_{e \in \pi} l_e$; thus, light paths contain high-confidence edges.

Computing basis CPIMs. We start by constructing a basis set of CPIMs in which each CPIM (T_1, T_2, O) has the following properties: (i) O contains two edges $(a, a') \in H$ and $(b, b') \in H$; (ii) a and b are adjacent in G_1 (i.e., T_1 is the edge (a, b)); and (iii) a' and b' are adjacent in G_1 .

Expanding a basis CPIM. GraphHopper processes each CPIM in the basis set using the following iterative algorithm. Figure 1 displays these steps.

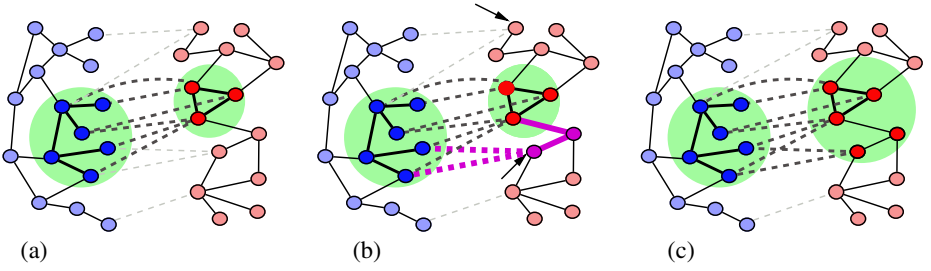


Fig. 1. An illustration of how GraphHopper expands a CPIM in iteration k . (a) A CPIM at the end of iteration $k - 1$. (b) In iteration k , GraphHopper keeps the blue network in the CPIM fixed and expands the red network. Arrows mark the two nodes in the set P computed in Step (ii). The node v' found in Step (iii) is the lower of these two nodes. In Steps (iv) and (v), GraphHopper adds the thick magenta PPis and orthology edges to the CPIM. (c) The CPIM at the end of iteration k .

Let (T_1^1, T_2^1, O^1) be a basis CPIM. In iteration $k > 1$ (Figure II (a)), we construct a CPIM (T_1^k, T_2^k, O^k) such that $(T_1^{k-1}, T_2^{k-1}, O^{k-1})$ is a subgraph of (T_1^k, T_2^k, O^k) and $\phi(T_1^k, T_2^k, O^k) > \phi(T_1^{k-1}, T_2^{k-1}, O^{k-1})$. We also attempt to keep $q(T_1^k, T_2^k, O^k) - q(T_1^{k-1}, T_2^{k-1}, O^{k-1})$ as small as possible. We keep either T_1^{k-1} or T_2^{k-1} fixed and “expand” the other graph. Without loss of generality, we assume that $T_1^k = T_1^{k-1}$ and T_2^{k-1} is a subgraph of T_2^k in the following discussion. We construct (T_1^k, T_2^k, O^k) using the following steps:

- (i) We identify a set $P \subseteq V_2$ of nodes such that each node $v \in P$ is not a node in T_2^{k-1} and is connected by an edge in H to at least one node in T_1^k .
- (ii) For each node $v \in P$, we use Dijkstra’s algorithm to compute the lightest path π_v in G_2 that connects v to T_2^{k-1} , i.e., for each node $u \in T_2^{k-1}$, we compute the lightest path between u and v in G_2 , and set π_v to be the lightest of these paths.
- (iii) We find the node v' in P such that $\pi_{v'}$ is the lightest among all paths computed in the previous step.
- (iv) We set T_2^k to be the union of T_2^{k-1} and $\pi_{v'}$ (Figure II (b)).
- (v) We set O^k to be the union of O^{k-1} and the set of edges in H incident on v' and a node in T_1^k (Figure II (b)).
- (vi) We compute $\phi(T_1^k, T_2^k, O^k)$. If $\phi(T_1^k, T_2^k, O^k) > \phi(T_1^{k-1}, T_2^{k-1}, O^{k-1})$, we go to Step (i) and expand (T_1^k, T_2^k, O^k) while keeping T_2^k fixed (Figure II (c)). Otherwise, we proceed to the next basis CPIM.

We provide the rationale for these steps. To expand the CPIM $(T_1^{k-1}, T_2^{k-1}, O^{k-1})$ after setting $T_1^k = T_1^{k-1}$, we first identify the set P of nodes in G_2 that do not belong to T_2^{k-1} but are orthologs of nodes in T_1^k (Step (ii)). Each node in P is a candidate that we can add to T_2^{k-1} in order to construct T_2^k . However, such a node $v \in P$ may not be adjacent to any node in T_2^{k-1} , as displayed in Figure II (b). Since our goal is to keep $q(T_1^k, T_2^k, O^k) - q(T_1^{k-1}, T_2^{k-1}, O^{k-1})$ as small as possible, we would like to connect v to T_2^{k-1} using the edges with the highest

possible confidence in G_2 . A natural candidate for this set of edges is the lightest path π_v connecting v to T_2^{k-1} , where this minimum is taken over the set of lightest paths connecting v to each node in T_2^{k-1} . Therefore, for each node v in P , we compute the lightest path π_v by which we can connect v to T_2^{k-1} using only edges in G_2 (Step (iii)). In Steps (iii) and (iv), we add that path π'_v to T_2^{k-1} that is lightest among all the paths computed i.e., $v' = \arg \min_{v \in P} |\pi_v|$. After computing T_2^k , we set O^k to be the subgraph of H induced by the nodes in T_1^k and T_2^k by adding the edges in H that are incident on v' and any node in T_1^k (Step (v)); by construction, no node in π'_v other than v' is connected by an edge in H to a node in T_1^k . This step completes the construction of (T_1^k, T_2^k, O^k) . Finally, in Step (vi), we continue expanding (T_1^k, T_2^k, O^k) if its conservation score is greater than $\phi(T_1^{k-1}, T_2^{k-1}, O^{k-1})$. Otherwise, we stop the iteration and move on to the next basis CPIM. By induction, the graphs T_1^k , T_2^k and $T_1^k \cup T_2^k \cup O^k$ are connected. Note that $q(T_1^k, T_2^k, O^k)$ implicitly plays a role in the expansion: since both the unreliability score of a CPIM and the lightness of a path are defined as the sum of the l_e values of the edges that appear in the CPIM or the path, by choosing to add the lightest path $\pi_{v'}$ to T_2^k , we are attempting to minimize $q(T_1^k, T_2^k, O^k) - q(T_1^{k-1}, T_2^{k-1}, O^{k-1})$.

Merging CPIMs. Following Sharan et al. [9], we compute the statistical significance of a CPIM by comparing its conservation score to the distribution of conservation scores of CPIMs found by GraphHopper in random PPI and orthology networks with the same degree distributions as G_1 , G_2 , and H . We retain CPIMs with p -value at most 0.05. The remaining CPIMs may have considerable overlap. We merge CPIMs by modifying the procedure used by Sharan et al. [9]. For each CPIM C , we compute all the biological functions it is enriched in using Fischer’s exact test and note the function f_C that is most enriched (has smallest p -value) in C . Let F be the set of all such most-enriched functions. For each function $l \in F$, we compute a CPIM C_l as the union of all CPIMs C for which $l = f_C$, i.e., $C_l = \bigcup_{l=f_C} C$. We report results for these CPIMs. Note that this method (i) does not require us to provide a cutoff on the overlap of two CPIMs that should be merged, (ii) allows merged CPIMs to share both proteins and interactions, and (iii) may yield disconnected CPIMs.

Remarks. There is considerable scope for variation in our algorithm. For instance, we can define the conservation and unreliability scores differently, combine the two scores, use simulated annealing-like techniques to optimize these scores, or focus on optimizing the unreliability score instead of the conservation score. We have experimented with a number of such choices (data not shown) and found that the algorithm presented consistently achieves good results.

3 Results

3.1 Comparison to NetworkBlast and Match-and-Split

We compared GraphHopper to NetworkBlast [9], a state-of-the-art method for computing CPIMs from alignment networks, and to Match-and-Split [6], which

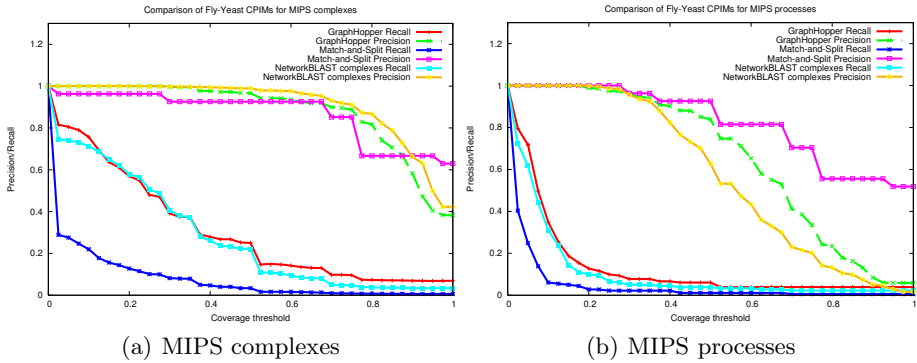


Fig. 2. Comparisons of GraphHopper, Match-and-Split, and NetworkBLAST

that like GraphHopper finds CPIMs by keeping the two PPI networks separate. Both papers used the same fly and yeast datasets. We downloaded these datasets and the results obtained by these algorithms from the supplementary websites accompanying the respective papers. We ran GraphHopper on exactly the same fly and yeast datasets. We used the procedure suggested by Narayanan and Karp [6] to compare the algorithms. We computed fly-yeast CPIMs and considered only the yeast sub-network in each CPIM. We considered two sets of gold standard modules defined by MIPS process annotations and by MIPS complex annotations for yeast genes [10]. We defined one set S of proteins as being *covered by* another set S' if $|S \cap S'|/|S| \geq t$, for a threshold $0 \leq t \leq 1$.¹ For a given value of t , we measured the *precision* of an algorithm as the fraction of computed CPIMs covered by at least one gold standard module and the *recall* of the algorithm as the fraction of gold standard modules covered by at least one computed CPIM. For each algorithm, we plotted precision and recall at different values of t . Precision and recall are both equal to 1 for $t = 0$. Both measures decrease monotonically with increasing t .

Figure 2 displays these results. For MIPS complexes, all three algorithms have comparable precision across almost the entire range of the coverage threshold. However, GraphHopper and NetworkBLAST have better recall than Match-and-Split. Match-and-Split achieves the best precision for MIPS processes. For this gold standard, GraphHopper has better precision and recall than NetworkBLAST. We obtain results similar to Figure 2(b) for KEGG processes (data not shown). These results are based on 766 GraphHopper CPIMs, 835 NetworkBLAST modules, and 27 Match-and-Split modules. Thus, Match-and-Split computes many fewer modules than the other two algorithms. On average, Match-and-Split modules are much smaller than those computed by GraphHopper and NetworkBLAST. Thus, GraphHopper is competitive with and sometimes outperforms Match-and-Split in spite of computing a much larger number of CPIMs.

Comparison of topological diversity. To underscore the diversity of the topologies of the CPIMs computed by GraphHopper, we performed another comparison

¹ Narayanan and Karp only considered $t = 0.5$.

of the three algorithms. We partitioned each computed CPIM into its two species-specific components and computed the ratio of the number of proteins in the larger component and the numbers of proteins in the smaller component. We observed that both Match-and-Split and NetworkBLAST computed CPIMs for which these ratios were between one and two. In contrast, GraphHopper computed a number of CPIMs with ratio at least 2.5. An example is a CPIM containing 4 yeast and 11 fly proteins that is enriched in the cellular component “myosin” (7.8×10^{-7})². Myosin is a protein complex that functions as a molecular motor, using the energy of ATP hydrolysis to move actin filaments or cargo on actin filaments. This CPIM may suggest how interactions between myosin proteins have evolved from single-celled to multi-cellular organisms.

3.2 Datasets

In the rest of this section, we present results obtained by GraphHopper on human, fly, and baker’s yeast protein interaction networks. We obtained 31610 interactions between 7393 human proteins from the IDSERVE database [12]. We removed interactions in the IDSERVE data that were obtained by transfer from lower eukaryotes based on sequence similarity. We also included 3270 human interactions derived using large scale yeast two-hybrid experiments from Stelzl et al. [13], and 6726 human PPIs from Rual et al. [14]. Overall, this human PPI network contained 7787 proteins and 30703 interactions and represents interactions from a diverse variety of sources. From the Database of Interacting Proteins [15], we collected 22004 interactions between 7350 fly proteins and 15317 interactions between 5019 yeast proteins. To find orthologous pairs of proteins, we ran BLASTP on a database containing all human, fly, and yeast protein sequences and retained only bidirectional hits with E -values less than 10^{-7} . We gathered functional annotations from the Gene Ontology (GO).

3.3 A Global Map of Human-Fly CPIMs

We find 265 human-fly CPIMs enriched in 969 functions, 149 human-yeast CPIMs enriched in 784 functions, and 34 fly-yeast CPIMs enriched in 273 functions. 161 functions enriched in all three comparisons span a diverse range of cellular activities including biological process such as cytokinesis, protein metabolism, and reproduction; molecular functions including microfilament motor activity, GTPase activity, and cyclin-dependent protein kinase activity; and cellular components such as the microtubule and the endoplasmic reticulum.

We find 163 functions enriched exclusively in human-fly CPIMs. Many of these functions are unique to multi-cellular organisms, for example, cell-matrix adhesion (2×10^{-13}), tissue development (3×10^{-6}), cell differentiation (1.2×10^{-13}), and ectoderm development (1.6×10^{-10}). Several CPIMs are enriched in functions related to sexual reproduction, such as embryonic development (4.2×10^{-11}), germ-line stem cell division (6.7×10^{-9}) and ovarian follicle cell development

² Numbers in parentheses are Bonferroni-corrected p -values of functional enrichment.

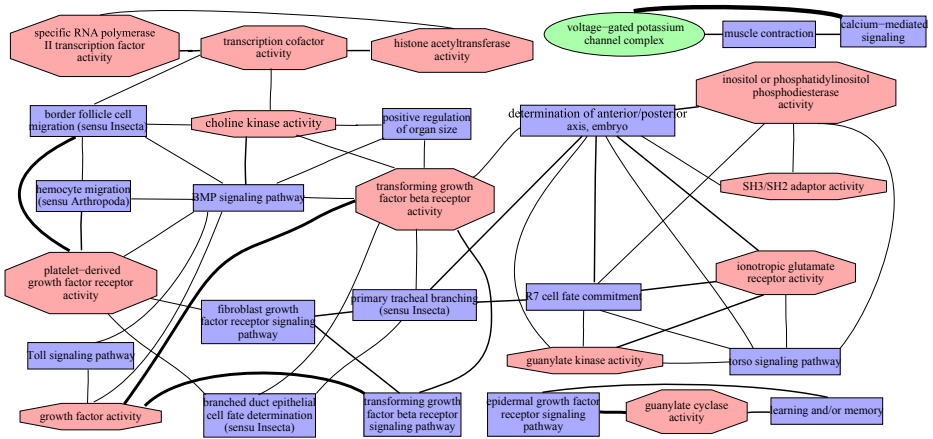


Fig. 3. A network of functions enriched in human-fly CPIMs. To construct this image, we associate each human-fly CPIM with the GO function most enriched in that CPIM, restricting our attention to functions with p -value of 10^{-4} or better. We ignore a CPIM c if there is another CPIM c' such that the GO function associated with c is an ancestor of the GO function associated with c' . We construct a network where each node is a CPIM and an edge connects two nodes if their CPIMs overlap. The thickness of an edge in Figure 3 represents the degree of overlap in terms of fraction of shared proteins (the ratio of the size of the intersection to the size of the union). We discard CPIMs that had at most 5% similarity to every other CPIM. We visualize the resulting network using the Graphviz package [16]. Blue rectangles are GO biological processes, red octagons are GO molecular functions, and green ellipses are GO cellular components.

(9.03×10^{-8}). A number of CPIMs are related to the development of the nervous system, for example, axon guidance (0.03), dopamine receptor activity (5.6×10^{-17}), and voltage-gated potassium channel complex (8.6×10^{-6}).

Figures 3 and 4(a) display connected components of a global network of functions enriched only in human-fly CPIMs and connections between these CPIMs. The largest component of the network in Figure 3 spans a diverse set of processes and functions, of which many are unique to multi-cellular organisms. The connected component of the human-fly CPIM network in Figure 4(a) connects five GO biological processes (negative regulation of fusion cell fate specification, Notch signaling pathway, ommatidial rotation, R3/R4 cell differentiation, and regulation of R8 spacing) to three GO molecular functions (calcium ion binding, extracellular matrix structural constituent, and transmembrane receptor protein phosphatase activity). Three of these CPIMs describe processes involved in eye development in fly (ommatidial rotation, R3/R4 cell differentiation, and regulation of R8 spacing). These CPIMs are connected by a module enriched in “transmembrane receptor protein phosphatase activity.” Tyrosine protein phosphatases such as *Dlar* play a critical role in controlling motor neuron guidance and targeting R cells correctly to different layers of the fly compound eye [17]. Other CPIMs are enriched in the molecular functions “calcium ion binding,” and “extracellular matrix structural constituent,” which reflect the roles played by the human proteins in these

4 Discussion

Earlier methods [2,4,9] for computing CPIMs have succeeded in detecting complexes and pathways conserved between two or more species. For instance, these models assume a pathway-like [2] or a complex-like [4] interaction structure between all the proteins in a module. Methods that integrate multiple PPI networks into a single alignment graph [2,3,4] are likely to compute CPIMs where the constituent protein interaction modules have similar topologies. The Graemlin algorithm allows the user to specify the topology of the protein interaction modules to be aligned; however, both modules in a CPIM must have similar topologies. An advantage that some previous methods have over GraphHopper is that they can simultaneously align more than two PPI networks [20,7,9].

CPIMs found by GraphHopper have a wider range of topologies than those computed by other methods. For example, the CPIM in Figure 4(b) maps the integrin complex in fly (6 proteins, 5 interactions) to a much larger and more dense human integrin network (32 proteins, 85 interactions). Such CPIMs are useful for capturing the increased diversity and complexity of a module of proteins in a higher eukaryote. This CPIM also demonstrates GraphHopper's ability to align a clique-like module with a module like a star graph.

We conclude by noting that CPIMs have been used to transfer protein functional annotations from one organism to another [6,9]. Most predicted functions correspond to fundamental processes of life. Our results, e.g., the suggested evolution of eye development proteins in fly to human proteins involved in blood clotting and vascular development, indicate the transfer of function for processes unique to multi-cellular organisms requires new techniques.

Acknowledgments. Grants from the ASPIRES program and the Institute for Critical Technology and Applied Science at Virginia Tech supported this research. We thank Vandana Sreedharan for many useful discussions.

References

1. Sharan, R., Ideker, T.: Modeling cellular machinery through biological network comparison. *Nat. Biotechnol.* 24(4), 427–433 (2006)
2. Kelley, B.P., Sharan, R., Karp, R.M., Sittler, T., Root, D.E., Stockwell, B.R., Ideker, T.: Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc. Natl. Acad. Sci. U S A* 100(20), 11394–11399 (2003)
3. Koyuturk, M., Kim, Y., Topkara, U., Subramaniam, S., Szpankowski, W., Grama, A.: Pairwise alignment of protein interaction networks. *J. Comput. Biol.* 13(2), 182–199 (2006)
4. Sharan, R., Ideker, T., Kelley, B.P., Shamir, R., Karp, R.M.: Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. In: RECOMB 2004: Proceedings of the eighth annual international conference on Computational molecular biology, pp. 282–289. ACM Press, New York (2004)

5. Suthram, S., Shlomi, T., Ruppin, E., Sharan, R., Ideker, T.: A direct comparison of protein interaction confidence assignment schemes. *BMC Bioinformatics* 7, 360 (2006)
6. Narayanan, M., Karp, R.M.: Comparing Protein Interaction Networks via a Graph Match-and-Split Algorithm. *J. Comput. Biol.* 14(7), 892–907 (2007)
7. Koyuturk, M., Kim, Y., Subramaniam, S., Szpankowski, W., Grama, A.: Detecting conserved interaction patterns in biological networks. *J. Comput. Biol.* 13(7), 1299–1322 (2006)
8. Gandhi, T.K., et al.: Analysis of the human protein interactome and comparison with yeast, worm and fly interaction datasets. *Nat. Genet.* 38(3), 285–293 (2006)
9. Sharan, R., Suthram, S., Kelley, R.M., Kuhn, T., Mccuine, S., Uetz, P., Sittler, T., Karp, R.M., Ideker, T.: From the cover: Conserved patterns of protein interaction in multiple species. *Proc. Natl. Acad. Sci. U S A* 102(6), 1974–1979 (2005)
10. Guldener, U., Munsterkotter, M., Oesterheld, M., Pagel, P., Ruepp, A., Mewes, H.W., Stumpflen, V.: MPact: the MIPS protein interaction resource on yeast. *Nucleic Acids Res.* 34, D436–D441 (2006)
11. Goldberg, D.S., Roth, F.P.: Assessing experimentally derived interactions in a small world. *Proc. Natl. Acad. Sci. U S A* 100(8), 4372–4376 (2003)
12. Ramani, A.K., Bunescu, R.C., Mooney, R.J., Marcotte, E.M.: Consolidating the set of known human protein-protein interactions in preparation for large-scale mapping of the human interactome. *Genome Biol.* 6(5), R40 (2005)
13. Stelzl, U., et al.: A human protein-protein interaction network: a resource for annotating the proteome. *Cell* 122(6), 957–968 (2005)
14. Rual, J., et al.: Towards a proteome-scale map of the human protein-protein interaction network. *Nature* 437(7062), 1173–1178 (2005)
15. Xenarios, I., Rice, D.W., Salwinski, L., Baron, M.K., Marcotte, E.M., Eisenberg, D.: DIP: the database of interacting proteins. *Nucleic Acids Res.* 28(1), 289–291 (2000)
16. Gansner, E.R., North, S.C.: An open graph visualization system and its applications to software engineering. *Software – Practice and Experience* 30(11), 1203–1233 (2000)
17. Araujo, S.J., Tear, G.: Axon guidance mechanisms and molecules: lessons from invertebrates. *Nat. Rev. Neurosci.* 4(11), 910–922 (2003)
18. Giot, L., et al.: A protein interaction map of drosophila melanogaster. *Science* 302(5651), 1727–1736 (2003)
19. Peri, S., et al.: Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Res.* 13(10), 2363–2371 (2003)
20. Flannick, J., Novak, A., Srinivasan, B.S., McAdams, H.H., Batzoglou, S.: Graemlin: general and robust alignment of multiple large interaction networks. *Genome Res.* 16(9), 1169–1181 (2006)

The 2-Interval Pattern Matching Problems and Its Application to ncRNA Scanning

Thomas K.F. Wong¹, S.M. Yiu¹, T.W. Lam¹, and Wing-Kin Sung²

¹ Department of Computer Science, The University of Hong Kong, Hong Kong
{kfwong, smyiu, twlam}@cs.hku.hk

² School of Computing, National University of Singapore, 3 Science Drive 2,
Singapore 117543
ksung@comp.nus.edu.sg

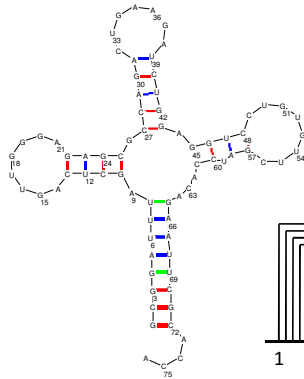
Abstract. This paper focuses on the 2-Interval pattern matching problem for $\{<, \sqsubset\}$ -structured pattern and applies it on scanning for the ncRNAs without pseudoknots. Vialette [6] gave an $O(mn^3 \log n)$ time solution to the problem, where m, n are the number of intervals in the pattern and the given 2-interval set. This solution however is not practical for scanning the secondary structure in a genome-wide or chromosome-wide scale. In this paper, we propose an efficient algorithm to solve the problem in $O(mn \log n)$ time. In order to capture more characteristics of the secondary structures of ncRNA families, we define a new problem by considering the distance constraints between the intervals and we can still solve it without increasing the time complexity. Experiment showed that the method to the new defined problem can result in much fewer false positives. Moreover, if we assume the only possible base pairs are $\{(A,U), (C,G), (U,G)\}$ which are the case for RNA molecule, we can further improve the time complexity to $O(mq)$, where q is the length of the input RNA sequences. From the experiment, our new method requires a reasonable time (2.5 min) to scan the whole chromosome for an ncRNA family.

1 Introduction

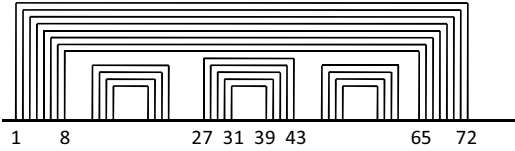
RNA can be regarded as a sequence of $\{A, C, G, U\}$ characters (called bases). Bases may bind together to form pairs. The base pairs of an RNA molecule define its secondary structure (see Figure 1(a) for an example). The functionality of an RNA molecule is closely related to its secondary structure. Certain substructures (e.g. a hairpin loop) in a secondary structure may be critical for the molecule to carry out its biological function. So, matching structural patterns in a secondary structure is an important subject in RNA study. For example, an important application is to check if a given RNA sequence contains a substring which belongs to an ncRNA (non-coding RNA) family [1,2]. We refer this problem as the ncRNA scanning problem (see [3-5]). Members in the same ncRNA family are believed to be conserved in the secondary structures, so to solve the ncRNA scanning problem, one can compare the secondary structure of the substrings in the RNA sequence with the consensus secondary structure of the members in the ncRNA family.

In a typical secondary structure of an RNA molecular, consecutive bases may bind to another set of consecutive bases forming a stacking pair. Consecutive bases can be modeled as an interval and a stacking pair can be modeled as a 2-interval. For example, in Figure 1(b), bases 1 to 8 form a stacking pair with bases 65 to 72. Bases 1 to 8 can be modeled as the interval $I = [1..8]$ while bases 65 to 72 can be modeled as the interval $J = [65..72]$, thus this stacking pair can be modeled as the 2-interval (I, J) .

(a) Secondary structure for phenylalanyl-tRNA



(b) A schematic view showing how 2-intervals relate to the secondary structures (using phenylalanyl-tRNA as an example). The stacking pairs between bases 1 to 8 and bases 65 to 72 can be modeled as the 2-interval (I, J) with $I = [1..8]$ and $J = [65..72]$; similarly, the stacking pairs between bases 27 to 31 and bases 39 to 43 can be treated as the 2-interval (I', J') with $I' = [27..31]$ and $J' = [39..43]$.



(c) The enhanced representation to capture distance between two stacking pairs and size of a hairpin loop
 (((((((. ((((.....)))) . ((((.....)))) . (((.....)))) ...)))))))

Fig. 1.

The representation of 2-intervals for stacking pairs is not new and pattern finding problems on 2-interval sets were proposed to study RNA. In particular, the 2-interval pattern matching problem was originally studied by Vialette [6], then subsequently in [7-9]. In this paper, we consider an important version of the 2-interval pattern matching problem which relates to the secondary structures of RNA molecules without pseudoknots. The problem is defined as follows.

Consider a set D of 2-intervals. A subset of D is said to be $\{<, \subset\}$ -comparable if every two 2-interval (I_1, J_1) and (I_2, J_2) in D have one of the followings relations: (i) $<$ (precedence), if $I_1 < J_1 < I_2 < J_2$; (ii) \subset (nesting), if $I_1 < I_2 < J_2 < J_1$. Let P be a string of balanced parenthesis. Note that every pair of balanced parenthesis in P also satisfies the precedence and nesting relations and P is called an $\{<, \subset\}$ -structured pattern. We say that P occurs in D if there exists a precedence and nesting relation preserving bisection between a subset of $|P|/2$ 2-intervals in D and the set of $|P|/2$ balanced parenthesis in P . The 2-interval pattern matching problem is to check if P occurs in D .

The solution for this problem can be applied to solve the ncRNA scanning problem as follows. The pattern P can be the consensus secondary structure of an ncRNA family and D is the given RNA sequence, if P occurs in D , it implies that the occurrences of P (substrings in D) may be a member of the ncRNA family. In practice,

researchers would like to have a fast scanning algorithm so that potential members (substrings) in D can be identified and further verified. A good solution should be fast, scalable for long sequences with high sensitivity (that is, real answers should not be missed) and relative small percentage for false positives. Vialette [6] gave an $O(m|D|^3 \log |D|)$ time solution to the problem, where m is the number of intervals in P . This solution is not practical to scan for the secondary structure in a genome-wide or chromosome-wide scale. In this paper, we show that this problem can be solved in $O(m |D| \log |D|)$ time.

Note that the 2-interval pattern matching problem defined does not capture the size of a loop (i.e. the distance of unpaired bases inside hairpin, internal-loop, bulge or multi-loop) nor the number of base pairs inside the stacking-pair region. These characteristics are important for the functionality of RNA molecules (for example, the size of a hairpin loop may affect the functions of an RNA molecule). Hence, using Vialette's definition, we may get a lot of false positives when using it to scan a RNA sequence for members in an ncRNA family. To handle this issue, we extend the problem as follows.

First, we enhance the representation of a secondary structure of an RNA molecule to include the size of all the loops such as hairpin, internal-loop, bulge or multi-loop. Also, the number of base pairs in a stacking pair will also be included in the representation. Figure 1(c) shows the enhanced representation. Each base pair in a structure is represented by a pair of parenthesis as a 2-interval. The number of unpaired bases in between is represented by the same number of dots. In real applications, the size of a hairpin loop and the distance between stacking pairs as well as the number of base pairs in a stacking pair may vary. These numbers usually represent the minimum requirement to be satisfied. Values of these numbers can be estimated from the consensus structure of an ncRNA family. Secondly, to match a substring in D with P , we limit the length of the substring to be less than or equal to a threshold len_{max} . By introducing these distance (size) constraints, we have set a more stringent requirement for a substring D to be similar to P . We then introduce the *2-interval pattern matching problem with distance constraints* (to be formally defined in Section 2) to model these new requirements.

According to our knowledge, this newly formulated problem has not been addressed in the literature. We show that this problem can also be solved in same $O(m |D| \log |D|)$ time by generalizing our solution to the 2-interval pattern matching problem. Moreover, if we assume the only possible base pairs are $\{(A,U), (C,G), (U,G)\}$ which are the case for RNA molecule, we can further improve the time complexity to $O(m q)$, where q is the length of the input RNA sequences.

We apply our solution to solve the ncRNA scanning problem and show that it is feasible for scanning a long sequence such as a chromosome within a reasonable amount of time (2.5 mins) for an ncRNA family. Also, we demonstrate that using our extended problem definition, we capture the characteristics of the secondary structures of ncRNA families better than the original definition, thus having a lot fewer false positives. As far as we know, most of the existing solutions (see for examples, [10-12]) for ncRNA scanning problem are sequence-based only. Our work represents another direction which makes use of the structural information (see also [13]). In [13], the authors modeled the application as a 2D (0,1) image pattern matching problem and showed that the solution is effective with few false positives based on

random generated RNA sequences. The performance of their solution in real data sets is still an unknown. A comparison on their method will be performed later.

2 Problem of Definition

The *2-Interval Pattern problem* is defined as follows:

Given a set of 2-intervals D and a $\{<, \subset\}$ -structured pattern P represented by a string of balanced parenthesis over the alphabet $\Sigma = \{ '(', ')', \cdot \}$, we would like to answer whether there exists a subset of D such that it is exactly the same as P . As shown in figure 2, in the other words, the question is: can we obtain pattern P by deleting all but $|P|/2$ 2-intervals from D ?

Before we define the second problem, let us define the *space-support pattern*. The space-support pattern includes not only the characters '(' and ')', but also a dot '.'. The space which is represented by a dot indicates there is at least a unit-distance between the adjacent characters. For example, a pattern $P = "(() \cdot \cdot ())"$ represents a pattern with two 2-intervals and at least 2-units distance between them. Note that a dot can appear in any position in P except the first and the last position. Also, if all of the dots are removed from P , then P will become a $\{<, \subset\}$ -structured pattern.

The *2-Interval Pattern problem with distance constraints* is defined as follows:

Given a set of 2-intervals D and a space-support pattern P , check whether there exists a subset of D such that (1) it is exactly the same as P which is the $\{<, \subset\}$ -structured pattern when all of the dots are removed from P ; (2) the distance between every two consecutive intervals is at least the distance indicated by the number of dots inside P . The idea is illustrated in Figure 3.

3 Method

3.1 The 2-Interval Pattern Matching Problem for $\{<, \subset\}$

Consider a set D of 2-intervals and a $\{<, \subset\}$ -structured query pattern P . The 2-Interval Pattern Matching Problem asks if P occurs in D . This section describes a dynamic programming algorithm to solve this problem in $O(mn \log n)$ time where m is the length of pattern P and n is the number of 2-intervals in the set D .

We model the $\{<, \subset\}$ -structured pattern P as a string of balanced parenthesis over the alphabet $\Sigma = \{ '(', ')', \cdot \}$. Hence, P satisfies the following property.

Lemma 1. Given the $\{<, \subset\}$ -structured pattern P represented as a balanced parenthesis, P can always be decomposed in one of the following ways: (1) $P = VW$ where V and W are two (non-empty) disjoint secondary structure patterns and $V = (Y)$ where Y is also a secondary structured pattern (which may be empty); (2) $P = (W)$ where $()$ denotes the outermost 2-interval I of P and W is the $\{<, \subset\}$ -structured pattern (which may be empty) obtained from P by deleting I .

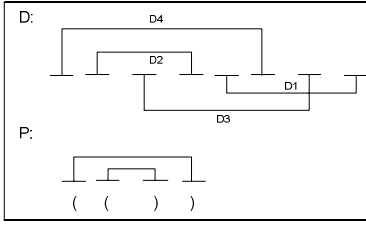


Fig. 2. The set of 2-intervals $D = \{D1, D2, D3, D4\}$. A $\{<, \subset\}$ -structured pattern $P = "(())"$. P occurs in D and P is representative of the subset $\{D2, D4\}$.

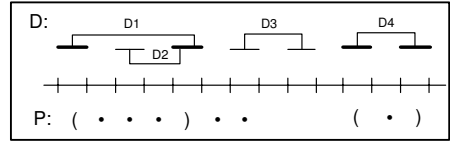


Fig. 3. The set of 2-intervals $D = \{D1, D2, D3, D4\}$. A space-support pattern $P = "(. . .) . . (.)"$. P occurs in D and P is representative of the subset $\{D1, D4\}$. The distance (1) between the two intervals in $D1$ is 3, (2) between $D1$ and $D4$ is 5, and (3) two intervals in $D4$ is 1. All fulfills the minimum distance constraints indicated by P , which is 3 for (1), 2 for (2) and 1 for (3).

This decomposition in Lemma 1 enables us to design a dynamic programming algorithm to solve the problem. Before we describe the recursive formula for the dynamic programming, we need some definitions. For any interval I , we denote $left(I)$ be the left position of interval I while $right(I)$ be the right position of interval I . Let $L = \{I \mid (I, J) \in D\}$ and $R = \{J \mid (I, J) \in D\}$. The intervals inside L are sorted according to their left positions in ascending order and each interval I inside L is labeled according to the sorted order (i.e. I_1, \dots, I_n). Also, the intervals inside R are sorted but according to their right positions in ascending order.

For any interval $I_i \in L$, we define $T(I_i, P)$ be the smallest position x such that P occurs inside the region $[left(I_i), x]$. Otherwise, set $T(I_i, P) = \infty$. For any $y \geq right(I_i)$, define $\varphi(I_i, y) = right(J_k)$ where J_k is the first interval $\in R$ such that $left(J_k) \geq y$ and $(I_i, J_k) \in D$. Define $\delta(x) = I_i$ where I_i is the first interval $\in L$ such that $left(I_i) \geq x$.

Note that if $T(I_n, P) \neq \infty$, P occurs in D . Below Lemma states the recursive formula for computing $T(I_i, P)$.

Lemma 2. For any interval I_i in L , depending on whether P is (1) $()$, (2) VW , or (3) (V) where V and W are some $\{<, \subset\}$ -structured patterns, the recursive formula for $T(I_i, P)$ is as follows.

$$\text{Case 1: When } P=(), T(I_i, P) = \min \begin{cases} T(I_{i+1}, P) \\ \varphi(I_i, right(I_i)) \text{ if exists} \\ \infty \text{ otherwise} \end{cases}$$

$$\text{Case 2: When } P = VW \text{ where } W \text{ is nonempty and } V=(U) \text{ where } U \text{ can be empty,}$$

$$T(I_i, P) = \begin{cases} T(\delta(T(I_i, V)), W) \text{ if } T(I_i, V) \neq \infty \text{ and } \delta(T(I_i, V)) \text{ exists} \\ \infty \text{ otherwise} \end{cases}$$

Case 3: When $P = (V)$ where V is nonempty,

$$T(I_i, P) = \min \begin{cases} T(I_{i+1}, P) \\ right(J_k) \text{ where } J_k = \varphi(I_i, T(I_m, V)) \text{ and } I_m = \delta(right(I_i)) \text{ if } I_m \text{ exists} \\ \infty \text{ otherwise} \end{cases}$$

Proof: For Case 1, there are two situations. In the first situation: I_i is not mapped with ‘(’, then $T(I_i, P) = T(I_{i+1}, P)$; In the second situation: I_i is mapped with ‘(’, then $T(I_i, P)$ is the first $J_k \in R$ such that $(I_i, J_k) \in D$. Thus $T(I_i, P) = \varphi(I_i, \text{right}(I_i))$.

For Case 2, let x be the smallest value such that the region $[\text{left}(I_i), x]$ contains V , i.e. $x = T(I_i, V)$. Let $I_k \in L$ be the first interval such that $\text{left}(I_k) \geq x$, i.e. $I_k = \delta(T(I_i, V))$. Then, $T(I_i, P) = T(I_k, W)$, resulting the smallest value of y such that the region $[\text{left}(I_k), y]$ contains W .

For Case 3, there are two situations. In the first situation: I_i is not mapped with ‘(’, then $T(I_i, P) = T(I_{i+1}, P)$; In the second situation: I_i is mapped with ‘(’, then let $I_m \in L$ be the first interval such that $\text{left}(I_m) \geq \text{right}(I_i)$, i.e. $I_m = \delta(\text{right}(I_i))$. Then, $x = T(I_m, V)$ is the smallest value such that the region $[\text{left}(I_m), x]$ contains V . Then locate the first $J_k \in R$ such that $\text{left}(J_k) \geq T(I_m, V)$ and $(I_i, J_k) \in D$, because J_k is mapped with ‘)’. Thus $J_k = \varphi(I_i, T(I_m, V))$. Therefore, $T(I_i, P) = \text{right}(J_k)$. \square

Based on Lemma 2, we compute $T(I_i, U)$ by scanning the intervals $I_i \in L$ from right to left and traversing the substructure U of P from inner to outer. Finally, we compute $T(I_i, P)$. If $T(I_i, P) \neq \infty$, then P occurs in D . Below two lemmas conclude the time complexity of the algorithm.

Lemma 3. For any 2-interval set D , using $O(n \log n)$ time, we can preprocess an indexing data-structure so that $\varphi(I_i, x)$ and $\delta(x)$ can be computed in $O(\log n)$ time.

Proof: The intervals in the list L and R can be sorted according to their starting positions and ending positions from left to right respectively using $O(n \log n)$ time. Regarding the data-structure for computing the values of $\varphi(I_i, x)$, for each $I_i \in L$, we can create a balanced binary tree to store all the starting positions of $J_k \in R$ such that $(I_i, J_k) \in D$. It takes $O(n \log n)$ time to build the data-structure. So, $\varphi(I_i, x)$ will takes $O(\log n)$ time to compute the value.

Similarly, for all $I_i \in L$ we can also create a balanced binary tree to store $(i, \text{left}(I_i))$ with key $\text{left}(I_i)$. It takes $O(n \log n)$ time to build the data-structure. Then, $\delta(x)$ can be computed in $O(\log n)$ time. \square

Lemma 4. For any 2-interval set D and $\{<, \subset\}$ -structured P , $T(I_i, P)$ can be computed in $O(mn \log n)$ time.

Proof: By Lemma 3, we have the data-structure for computing $\varphi(I_i, x)$ and $\delta(x)$. Together with Lemma 2, for any interval $I_i \in L$ and any substructure U of P , the value of $T(I_i, U)$ can be computed in $O(\log n)$ time. Since there are $m \times n$ entries of $T(I_i, U)$. In total, the running time is $O(mn \log n)$. \square

3.2 The 2-Interval Pattern Matching Problem with Distance Constraints

To solve the 2-interval pattern matching problem with distance constraints, we need to define an extra function:

Let $dot_front(P)$ = the number of consecutive dots just after the first '(' in P. For example,

If $P = "(\dots))"$, then $dot_front(P) = 2$, while if $P = "((\dots))"$, then $dot_front(P) = 0$.

Let $dot_end(P)$ = the number of consecutive dots just before the last ')' in P. For example,

If $P = "(\dots))"$, then $dot_end(P) = 1$, while if $P = "(\dots))"$, then $dot_end(P) = 0$.

Again, for any interval $I_i \in L$, we define $T(I_i, P)$ be the smallest position x such that P occurs inside the region $[\text{left}(I_i), x]$. Otherwise, set $T(I_i, P) = \infty$. Denote $“.*”$ be a sequence of ‘.’ of any length. We have the following lemma.

Lemma 5. For any interval I_i in L, depending on whether a space-support pattern P is (1) $(.*)$, (2) $V .* W$, (3) $(.*V.*)$ where V and W are some space-support patterns, the recursive formula for $T(I_i, P)$ is as follows.

Case 1: When $P = (.*)$, let $d = dot_front(P)$

$$T(I_i, P) = \min \begin{cases} T(I_{i+1}, P) \\ \varphi(I_i, \text{right}(I_i) + d) \text{ if exists} \\ \infty \text{ otherwise} \end{cases}$$

Case 2: When $P = V .* W$ where W is nonempty and $V = (U)$ where U can be empty, let $d = \text{the number of spaces between V and W}$.

$$T(I_i, P) = \begin{cases} T(\delta(T(I_i, V) + d), W) \text{ if } T(I_i, V) \neq \infty \text{ and } \delta(T(I_i, V)) \text{ exists} \\ \infty \text{ otherwise} \end{cases}$$

Case 3: When $P = (. * V .*)$ where V is nonempty and no dot is in the first and the end position of V, let $d_1 = dot_front(P)$ and $d_2 = dot_end(P)$

$$T(I_i, P) = \min \begin{cases} T(I_{i+1}, P) \\ \text{right}(J_k) \text{ where } J_k = \varphi(I_i, T(I_m, V) + d_2) \text{ and } I_m = \delta(\text{right}(I_i) + d_1) \\ \text{if } T(I_m, V) \neq \infty, \varphi(I_i, T(I_m, V)) \text{ exists and } \delta(\text{right}(I_i)) \text{ exists} \end{cases}$$

Proof: For Case 1, there are two situations. In the first situation: I_i is mapped with ‘(’, then $T(I_i, P) = T(I_{i+1}, P)$; In the second situation: I_i is mapped with ‘(’, then $T(I_i, P)$ is the first $J_k \in R$ such that the distance between $\text{right}(I_i)$ and $\text{left}(J_k)$ is at least d , and $(I_i, J_k) \in D$. Thus $T(I_i, P) = \varphi(I_i, \text{right}(I_i) + d)$.

For Case 2, let x be the smallest value such that the region $[\text{left}(I_i), x]$ contains V, i.e. $x = T(I_i, V)$. Let $I_k \in L$ be the first interval such that the distance between x and $\text{left}(I_k)$ is at least d , i.e. $I_k = \delta(T(I_i, V) + d)$. Then, $T(I_i, P) = T(I_k, W)$, resulting the smallest value of y such that the region $[\text{left}(I_k), y]$ contains W.

For Case 3, there are two situations. In the first situation: I_i is not mapped with ‘(’, then $T(I_i, P) = T(I_{i+1}, P)$; In the second situation: I_i is mapped with ‘(’, then let $I_m \in L$ be the first interval such that the distance between $right(I_i)$ and $left(I_m) \geq d_1$, i.e. $I_m = \delta(right(I_i) + d_1)$. Then, $x = T(I_m, V)$ is the smallest value such that the region $[left(I_m), x]$ contains V . Then locate the first $J_k \in R$ such that the distance between $T(I_m, V)$ and $left(J_k) \geq d_2$, and $(I_i, J_k) \in D$, because J_k is mapped with ‘)’. Thus $J_k = \varphi(I_i, T(I_m, V) + d_2)$. Therefore, $T(I_i, P) = right(J_k)$. \square

Lemma 6. For any 2-interval set D and a space-support pattern P , let I_m be the left interval of the rightmost 2-interval in D . Then, $T(I_m, P)$ can be computed in $O(m n \log n)$ time.

Proof: The number of comparisons in $dot_front(P)$ and $dot_end(P)$ is the number of consecutive dots just after the first ‘(’ in P plus 1 (i.e. $d_1 + 1$) and before the last ‘)’ plus 1 (i.e. $d_2 + 1$). Only $dot_front(P)$ is considered in case 1, while both in case 3. However, the input pattern for the next iteration will be shortened at least $d_1 + 1$ in case 1 and $d_1 + d_2 + 2$ in case 3. Therefore, for each $I \in L$, the total time required for $dot_front(P) + dot_end(P) = O(m)$. Therefore, the time required altogether for both functions is $O(m n)$.

Similarly, in case 2, the number of comparisons for computing the d value is $d + 1$. The pattern is then spitted into two and the sum of the length of the input patterns is decreased by d . Therefore, for each $I \in L$, the total time required altogether for computing the d value in case 2 is $O(2m) = O(m)$. So, the time required altogether is $O(m n)$.

Thus, the overall time complexity is still the same, which is $O(m n \log n + m n + mn) = O(m n \log n)$. \square

3.3 The Pattern Matching Problem with Distance Constraints on RNA

Consider an RNA sequence $S[1..q]$. Assume every individual base $S[i]$ is an interval (i.e. $left(S[i]) = i$ and $right(S[i]) = i + 1$). Suppose the set D of 2-intervals is $\{(S[i], S[j]) \mid S[i], S[j] \text{ form a base pairs } a-u, c-g, \text{ or } g-u\}$. In this case, the size of D can be as large as q^2 . Then, by Lemma 5, the pattern matching problem requires $O(m q^2 \log q)$ time. In this section, we show that the problem can be solved in $O(m q)$ time.

The idea is to replacing the indexing data-structure in Lemma 3 by that of Lemma 7. Then, it is easy to check that the time complexity is improved to $O(m q)$.

Lemma 7. For any 2-interval set D , using $O(q)$ time, we can preprocess an indexing data-structure so that $\varphi(I_i, x)$ and $\delta(x)$ can be computed in $O(1)$ time.

Proof: For any base σ in $\{a, c, g, u\}$, using $O(q)$ time, let $A_{\sigma}[x]$ be a length- q array which store the smallest value $y \geq x$ such that $S[y] = \sigma$. Then, $\varphi(a, x) = A_u[x]$, $\varphi(c, x) = A_g[x]$, $\varphi(g, x) = \min\{A_c[x], A_u[x]\}$, and $\varphi(u, x) = \min\{A_a[x], A_g[x]\}$. Hence, $\varphi(I_i, x)$ can be computed in $O(1)$ time.

.....((((((((((((.....)))))))))) ((.....((.....)).....) . (((.....)) .))((((.....))))

Fig. 6. The corresponding space-support patterns for the conserved structure in Figure 2

Given such consensus secondary structure, we represent its paired bases as a string of balanced parenthesis. Moreover, we also use the ‘dots’ to represent the unpaired bases. Figure 6 shows the corresponding secondary structure pattern. Given the secondary structure pattern P, we apply the method in Section 3.3 to scan the input sequences to locate possible members of the Rfam family. In our experiment, we assume the possible base-pairs in the input sequence are {a-u, c-g, u-g}. Furthermore, we assume the regions which contain the pattern P should be shorter than the maximum length of all the seed members. Next section presents the experimental result of the scanning.

5 Experimental Result

We selected five ncRNA families with conserved secondary structure for the experiment. In each family, a pattern is created according to the consensus secondary structure of the seed sequences as mentioned in Section 4. To evaluate the performance, all sequences in the family except the seed sequences are used. (Those sequences are denoted as full-members in the Rfam database.) The sensitivity is defined as the percentage of full-member sequences regarded as the members of the family by the method. On the other hand, we generated 100 random sequences for each family to test the false-positive rate (i.e. the percentage of random sequences regarded as the members of the family). Thus, higher the sensitivity and lower the false-positive rate indicate the better effectiveness of the method. We compared the performance of two approaches: (1) pattern with space support; (2) pattern without space support.

Table 1. Comparison between the methods by using pattern with and without dot support. Tp (true positive) is the # of full member sequences are regarded as the family member. Fp (false positive) is the # of random sequences are wrongly regarded as the family member. The corresponding rate is Tp rate (true-positive rate) and Fp rate (false-positive rate).

Family	Pattern with dot support						Pattern without dot support					
	Tp	total	Fp	total	Tp rate	Fp rate	Tp	total	Fp	total	Tp rate	Fp rate
RF00032	98	98	17	100	100%	17%	98	98	73	100	100%	73%
RF00111	15	15	1	100	100%	1%	15	15	100	100	100%	100%
RF00214	76	76	4	100	100%	4%	76	76	100	100	100%	100%
RF00383	7	7	20	100	100%	20%	7	7	100	100	100%	100%
RF00464	12	12	8	100	100%	8%	12	12	99	100	100%	99%
	average					10%	average					94%

As shown in Table 1, although both methods can identify all the full-member sequences as the family members, the method of using pattern with space support has significant improvement in the false-positive rate. The average false-positive rate among the five chosen families by using pattern with dot support is 10% while that of

by using pattern without dot support is 94%. The result indicates the considerable necessity of using dot-support pattern.

We also selected the family RF00111 and scanned through the whole human chromosome 14 to locate the possible members of the family. It took only 2.5 minutes under the machine with 2.4G CPU with 4G.

References

- [1] Griffiths-Jones, S., Bateman, A., Marshall, M., Khanna, A., Eddy, S.R.: Rfam: An RNA family database. *Nucleic Acids Research* 31(1), 439–441 (2003)
- [2] Noncoding RNA database, <http://biobases.ibch.poznan.pl/ncRNA>
- [3] Klein, R., Eddy, S.: RSEARCH: Finding homologs of single structured RNA sequences. *BMC Bioinformatics* 4(1), 44 (2003)
- [4] Wong, T., Chiu, Y.S., Lam, T.-W., Yiu, S.M.: A memory efficient algorithm for structural alignment of RNAs with embedded simple pseudoknots. In: *Proceedings of the 6th Asia-Pacific Bioinformatics Conference*, pp. 89–99 (2008)
- [5] Zhang, S., Hass, B., Eskin, E., Bafna, V.: Searching genomes for noncoding RNA using FastR. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2(4) (2005)
- [6] Vialette, S.: On the computational complexity of 2-interval pattern matching problems. *Theor. Comput. Sci.* 312(2-3), 223–249 (2004)
- [7] Blin, G., Fertin, G., Vialette, S.: New results for the 2-interval pattern problem. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) *CPM 2004. LNCS*, vol. 3109, pp. 311–322. Springer, Heidelberg (2004)
- [8] Chen, E., Yang, L., Yuan, H.: Improved algorithms for largest cardinality problem. *J. Comb. Optim.* 13(3), 263–275 (2007)
- [9] Crochemore, M., Hermelin, D., Landau, G.M., Vialette, S.: Approximating the 2-interval pattern problem. In: Brodal, G.S., Leonardi, S. (eds.) *ESA 2005. LNCS*, vol. 3669, pp. 426–437. Springer, Heidelberg (2005)
- [10] Nawrocki, E.P., Eddy, S.R.: Query-Dependent Banding (QDB) for faster RNA similarity searchers. *PLoS Computational Biology* 3(3), 540–554 (2007)
- [11] Weinberg, Z., Ruzzo, W.L.: Faster genome annotation of non-coding RNA families without loss of accuracy. In: *Proceedings of the 8th Annual International Conference on Computational Molecular Biology (RECOMB)* (2004)
- [12] Weinberg, Z., Ruzzo, W.L.: Sequence-based heuristics for faster annotation of non-coding RNA families. *Bioinformatics* 22(1), 35–39 (2006)
- [13] Yoon, B.-J., Vaidyanathan, P.P.: Fast structural similarity search of noncoding RNAs based on matched filtering of stem patterns. In: *IEEE conference on Signals, Systems and Computers (ACSSC 2007)*, pp. 44–48 (2007)

RNA Pseudoknot Folding through Inference and Identification Using TAG_{RNA}

Sahar Al Seesi, Sanguthevar Rajasekaran, and Reda Ammar

Computer Science and Engineering Department, University of Connecticut
{sahar, rajasek, reda}@enr.uconn.edu

Abstract. Studying the structure of RNA sequences is an important problem that helps in understanding the functional properties of RNA. After being ignored for a long time due to the high computational complexity it requires, pseudoknot is one type of RNA structures that has been given a lot of attention lately. Pseudoknot structures have functional importance since they appear, for example, in viral genome RNAs and ribozyme active sites. In this paper, we present a folding framework, TAG_{RNA}Inf, for RNA structures that support pseudoknots. Our approach is based on learning TAG_{RNA} grammars from training data with structural information. The inferred grammars are used to identify sequences with structures analogous to those in the training set and generate a folding for these sequences. We present experimental results and comparisons with other known pseudoknot folding approaches.

1 Introduction

Many new functional RNAs, such as miRNAs and tmRNAs [3] [20] [34] have been discovered in recent years. This resulted in speeding up RNA structural analysis and determination. Another factor that has led acceleration of RNA structural research is the rise of the RNA World Hypothesis [10] which suggests that the current DNA and protein world have evolved from an RNA based world. Analysis of the structures of RNA sequences is essential in understanding their functional properties. Consequently, it is imperative for creating new drugs and understanding genetic diseases [6] [24]. Computational methods can provide less expensive solutions to structure analysis than other methods such as nuclear magnetic resonance and x-ray crystallography.

Most RNA structure analysis research can be classified into thermodynamic or comparative approaches. Thermodynamic approaches use dynamic programming to compute the secondary structure with the minimum free energy (mfe) [13] [35]. These approaches use experimentally determined parameters for free energies. Comparative approaches are based on aligning a set of homologous sequences and computing the structure based on the alignment [8] [12]. Recently, a new approach for RNA structure analysis based on grammatical formalisms has emerged. This approach was inspired by David Searls work in the early 90's where he studied the linguistics of biological sequences [28]. He suggested the use of formal grammars as a tool to model and analyze DNA, RNA, and proteins. The use of grammars has attracted the attention of many researchers [26] [31] because it can model long range interactions. In

addition, grammatical models are concise and easy to understand representation of structures of sequence families.

A secondary structure for a sequence of length n is a list of base-pairs (bps) in the form (i, j) where $1 \leq i, j \leq n$. Two bps (i, j) and (k, l) are nested if $i < k$ and $l < j$. Two bps are crossing if $i < k$ and $j < l$. Pseudoknot is one type of RNA structures that exhibits crossing bps. It has been proven that predicting RNA structures with pseudoknots using free energy minimization is an NP-complete problem [17]. Also, pseudoknots cannot be modeled with Context Free Grammars (CFG) due to the crossing dependencies of their bps. Consequently, until recently, pseudoknots were ignored in RNA secondary structure analysis. Pseudoknot structures have functional importance since they appear, for example, in viral genome RNAs [19], ribozyme active sites [30], and tmRNA [34]. Among the available research in analyzing pseudoknot structures are the works of Akutsu [2], Dirks and Pierce [9], Rivas and Eddy [23], the iterated loop matching algorithm (ILM) by Ruan *et. al.* [25], and pknotsRG by Reeder and Giegerich [22].

One of the proposed grammatical models that support pseudoknots is TAG_{RNA}. TAG_{RNA} is a submodel of Tree Adjoining Grammars (TAG) [14]. It was proposed by Uemura *et. al.* [31]. They developed a parser for their model, and presented experimental results for using the model to fold RNA sequences with pseudoknot structures. Our solution is based on the TAG_{RNA} model.

Our solution is a grammatical inference approach to RNA structure analysis. Among the research that uses grammatical inference in bioinformatics are the works of Brazma *et. al.* [5], Laxminarayana *et. al.* [16], Takakura *et. al.* have published [29]. Brazma *et. al.* [5] have proposed an approach to discover simple grammars for families of biological sequences, using a subclass of regular grammars. On the use of grammatical inference to analyze RNA structures with Pseudoknots, Laxminarayana *et. al.* [16] presented an inference algorithm for Terminal Distinguishable Even Linear Grammars (TDELG), and they have shown how to use this algorithm in an Infer-Test model for the detection of a pseudoknot structure in an RNA sequence. They address the same problem we addressed in [1]. Takakura *et. al.* [29] use alignment data to infer probabilistic TAG_{RNA}. They use the inferred grammar to find new members of nc-RNA families. Sakakibara has published [27] in which he discusses the general merits of using grammatical inference in bioinformatics.

The use of grammatical inference to automate the grammar building step is essential in facilitating the use of grammatical formalism by biologists. Otherwise, the biologist will always be dependent on a grammar expert. In [1], we presented a grammatical inference engine for TAG_{RNA}. We also presented a structure identification framework, where the inferred grammars can be used to answer the question of whether an RNA input sequence exhibits a certain structure or not. In this paper, we present a modification of the framework which is capable of folding¹ an RNA sequence with identification as a first step in folding. We test our solution on RNA sequences from Pseudobase [4], Rfam [11], and the tmRNA database [34], and we compare our results with a representative subset of the available tools that are capable of folding RNA sequences including pseudoknots. We compare our results with ILM

¹ We use the terms structure prediction and folding interchangeably.

and, `pknotsRG`. `PknotsRG` is an algorithm for folding RNA sequences under the `mfe` model. It requires $O(n^4)$ time and $O(n^2)$ space. The `ILM` algorithm is based on the loop matching algorithm [18], and it also utilizes thermodynamic parameters. The worst case time complexity of `ILM` is $O(n^4)$ and the space complexity is $O(n^2)$. We also compare our results with `TAGRNA` which our solution is based on. The folding approach provided in [31] using `TAGRNA` is based on single generic grammar. Our approach is different because we infer specific grammars and use them to do identification and folding. `TAGRNA` has time and space complexity of $O(n^5)$ and $O(n^4)$ respectively.

2 TAG and TAG_{RNA}

Tree Adjoining Grammars (TAGs) were introduced by Joshi *et. al.* [14]. Uemura *et. al.* [31] defined a subclass of TAGs, `TAGRNA`, suitable for modeling RNA pseudoknot structures. In this section, we describe TAG and `TAGRNA`.

A Tree Adjoining Grammar (TAG) is defined to be a 5-tuple $(T \cup \{\epsilon\}, N, I, A, S)$, where T is a set of terminal symbols, N is a set of non-terminal symbols, ϵ is the empty string symbol, and S is the starting symbol. I and A are defined as follows:

I (initial trees): A finite set of finite trees with the internal nodes' labels belonging to $N \cup \{S\}$, the leaves' labels belonging to $T \cup \{\epsilon\}$, and the root is labeled with S .

A (auxiliary trees): A finite set of finite trees with the internal nodes' labels belonging to $N \cup \{S\}$, and the leaves' labels belonging to $T \cup \{\epsilon\}$ except one leaf node which has the same label as the root. This special leaf node is called a foot node.

$I \cup A$ constitutes the set of elementary trees. An operation called the adjoining operation can be used to compose two trees, resulting in a derived tree. The adjoining operation composes an auxiliary tree β with a foot node labeled X with any other tree α , elementary or derived, that has some internal node with the same label X . The adjoining operation works as follows: starting with the tree α , extract the sub-tree rooted at the internal node labeled with X (let that sub-tree be γ), and replace it with β . Then at the foot node of β , γ is reinserted. The adjoining operation is illustrated in Fig. 1. Let $T = \{ t : \exists i \in I \text{ s.t. } t \text{ can be derived from } i \}$, then $L(\text{TAG})$ consists of the yield of all the trees in T .

In [31], Simple Linear TAG (SLTAG) and Extended Simple Linear TAG (ESLTAG) are defined to be two subclasses of TAG with adjoining constraints [33]. In these two subclasses, the adjoining operation can occur only at internal nodes tagged

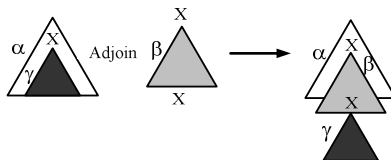


Fig. 1. The Adjoining Operation

with the symbol *, and the number of these nodes is restricted to one in SLTAG and two in ESLTAG. TAG_{RNA} is a sub-class of ESLTAG where only five types of elementary trees are allowed (Fig. 2). Each type of tree is responsible for a specific kind of branching or structural form that an RNA sequence can have.

3 The Structure Identification/Prediction Framework

In [1], we presented TAG_{RNA}Inf; an RNA structure identification framework. By structure identification we mean, given an RNA sequence, we answer the question of whether it exhibits a certain structure or not. TAG_{RNA}Inf, has a training phase in which a grammatical inference engine is fed with a positive training set with structural information. The inference engine will generate a grammar for each unique structure pattern in the sample. Then, the same sample along with a negative sample and the grammar(s) generated by the inference algorithm will go through an ESLTAG parser. For each input sequence the parser will output a score. We use the maximum number of base pairs as the scoring function. The scores will be the input to a threshold function inference module. This module infers a score threshold function $Th(l) = p$. A sequence s of size l is considered to have the RNA structure represented by a grammar G iff the parser accepts s under G , with score $p_s \geq p$. $Th(l)$ is a step function defined as follows:

$$Th(l) = p \quad , \quad i \leq l < j \quad (1)$$

The threshold function inference module infers a function that maximizes the sum of sensitivity and specificity for the training data using dynamic programming. The time and space complexity for inferring the threshold function are $O(n^3m^2)$ and $O(n^2m^2)$, where n is the maximum sequence size and m is maximum reported score for the training data set. While inferring the threshold function, this module also selects the most informative grammars. As mentioned earlier, the grammatical inference engine generates a grammar for each unique structure pattern it encounters. Here, nearly redundant grammars or grammars rarely used in the training set are eliminated. This enhances the time performance for the identification phase by reducing the number of grammars representing a training set. Furthermore, the number of grammars can be restricted to a preset maximum. For more details refer to [1].

The identification module consists of an ESLTAG parser and sets of inferred grammars coupled with their threshold functions. Each grammar set represents a certain structure. Depending on the training set fed to the inference engine, these structures could be as general as a pseudoknot structure, more specific as an H-type pseudoknot structure, or as specific as the structure of Antizyme RNA frameshifting stimulation element, for example. Given an input RNA sequence, the user can select to check it against a certain set of grammars. The identification module will answer the question of whether it belongs to the structure defined by this set of grammars.

The Identification/Prediction Phase

In this paper, we present a variant of TAG_{RNA}Inf which can be used to fold RNA sequences. In the new framework, the training phase remains unchanged. The Identification phase is replaced with an identification/prediction phase. In this phase, the

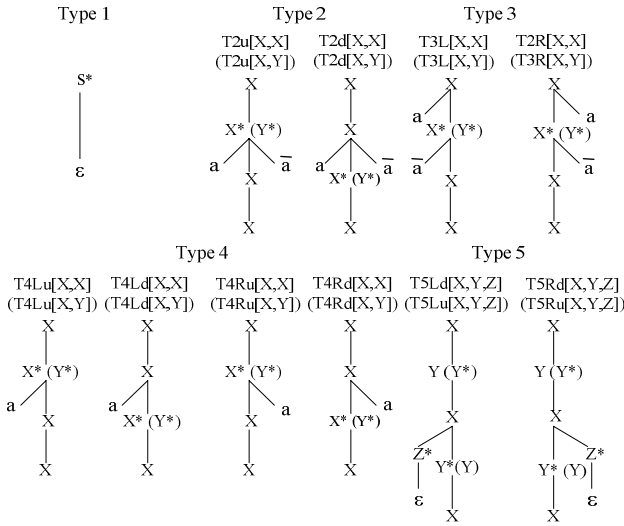


Fig. 2. TAG_{RNA}

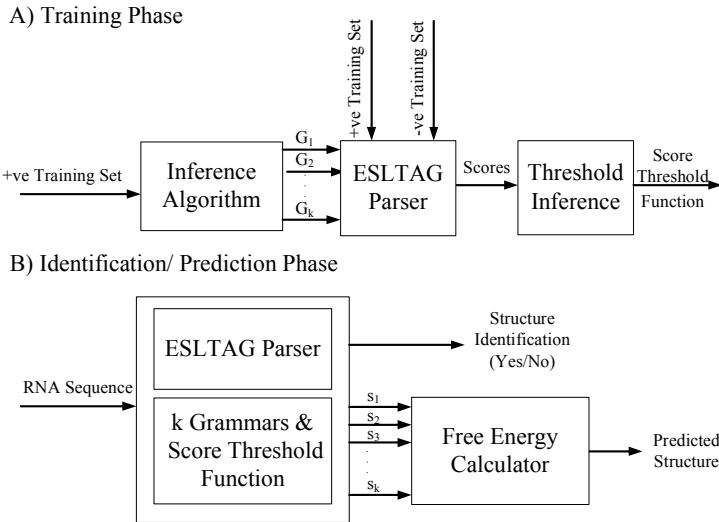


Fig. 3. TAG_{RNA}Inf : RNA structure identification/prediction framework

identification question is answered as before. Additionally, if the input sequence is identified to have the structure represented by the selected set of grammars, the sequence will be folded, and TAG_{RNA}Inf will output its structure. If the sequence was accepted by more than one grammar in the set, the structure resulting in minimum free energy is selected. To calculate the mfe for a certain structure, we use RNAeval tool from the Vienna suite [13]. RNAeval does not support pseudoknots. We

approximate the free energy of a pseudoknot by the sum of the free energies for its two stem-loops as calculated by RNAeval. If a sequence was accepted by the parser, resulting in a non-zero score, but was rejected by the threshold function, the user may choose to fold the sequence despite its rejection. However, the confidence level for this structure is not expected to be high. The framework is illustrated in Fig. 3.

The bottleneck for the computational complexity of our approach lies in the parser. We currently use an implementation of the SLTAG and ESLTAG parses described in [31]. If the set of grammars used do not have any TYPE5 trees (see Fig 2) the SLTAG parser is used; otherwise, the ESLTAG parser is used. The time complexity of the SLTAG and the ESLTAG parsers are $O(n^5)$ and $O(n^4)$ respectively. Both parsers have $O(n^4)$ space complexity.

4 Experimental Results

To test the accuracy of folding for TAG_{RNA}Inf, we evaluate the sensitivity and specificity of the predicted structures for a set of H-type pseudoknot sequences. The folding sensitivity and specificity are defined as follows:

$$\text{Folding_Sensitivity} = \frac{TP}{ref_bps} \quad \text{and} \quad \text{Folding_Specificity} = \frac{TP}{predicted_bps}$$

Where TP , ref_bps , and $predicted_bps$ are the number of correctly predicted bps, number of bps in the actual structure, and total number of predicted bps respectively.

For the training phase of this experiment, we used 105 H-type RNA sequences as the +ve training set and 107 non-pseudoknot sequences as the -ve training set. The +ve training data set was collected from Pseudobase [4], the tmRNA database [34], and Rfam database [11]. We arbitrarily selected sequences from tmRNA and extracted PK1, PK2, and PK4 from them. The negative training set was driven from the non-pseudoknot families in Rfam database, taking into consideration that the lengths of these sequences would be in the same range as the positive population. The +ve training set resulted in 6 grammars. The test set consists of 36 H-type pseudoknot sequences. The test set was driven from the same sources as the +ve training set. It includes 4 sequences from Rfam, 20 sequences from Pseudobase, and 12 from the tmRNA database.

We ran three comparative experiments on the test data. The first was an identification/structure prediction experiment. In this experiment, the test set was fed to the identification engine to check if the structure of each sequence belongs to any of the inferred grammars. The identification sensitivity and specificity for the training data set are 87.4 and 84.4 respectively. The sensitivity and specificity for identification are defined as:

Table 1. Folding sensitivity and specificity for the 31 sequenced accepted by the H-type pseudoknot grammars on TAG_{RNA}Inf

	Sensitivity	Specificity
ILM	69.1	67.7
pknotsRG (enf)	75.9	77.9
TAG _{RNA}	83.7	79.3
TAG _{RNA} Inf	83.4	87.4

Table 2. Folding sensitivity and specificity for the whole 36 sequence test set

	Sensitivity	Specificity
ILM	68.5	67.7
pknotsRG (enf)	75.5	77.0
TAG _{RNA}	80.0	75.5
TAG _{RNA} Inf	79.5	85.3

$$\text{Identification_Sensitivity} = \frac{TP}{TP + FN} \quad \text{and} \quad \text{Identification_Specificity} = \frac{TP}{TN + FP}$$

where TP , TN , FP , and FN are the number of true positives, true negatives, false positives, and false negatives respectively.

Out of the 36 input sequences, 31 were accepted and 5 were rejected. The structure generated by TAG_{RNA}Inf for the accepted 31 sequences were compared with the actual structures for these sequences, taken from the source databases, and *folding_sensitivity* and *folding_specificity* were calculated. We also generated structures for the same set of 31 sequences by ILM, pknotsRG, and TAG_{RNA}. pknotsRG was tested in enforce mode (enf), which enforces a pseudoknot in the predicted structure. Also, *Use extended helix plot score* option was set for ILM as recommended by the ILM website for single sequence structure prediction. All other options for all tools were set to their defaults. The grammars generated by TAG_{RNA}Inf have default minimum stem length of 2 and maximum bulge loop length of 2. Table1 lists the comparative results for this experiment. TAG_{RNA}Inf results in the best specificity with a big margin and the second best sensitivity after TAG_{RNA} with a very small difference. The high specificity of TAG_{RNA}Inf is expected because the grammars used for prediction are H-type pseudoknot grammars. Sequences whose structures are not expected to follow any of the inferred grammars are excluded in the identification phase, as will be illustrated further in the following two experiments. Figure 4 illustrate an example where TAG_{RNA}Inf gives more accurate structure prediction than ILM and pknotsRG.

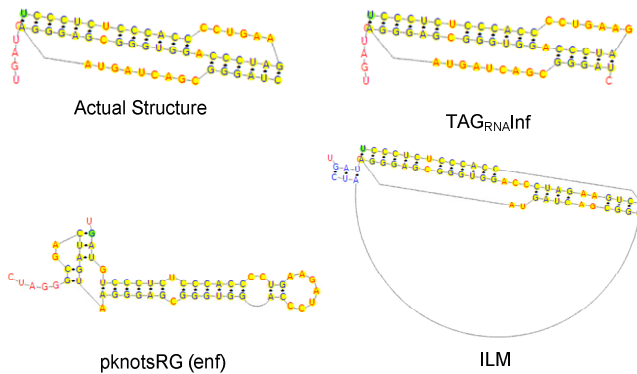


Fig. 4. Actual [11] and predicted structures for an Antizyme RNA frameshifting regulating sequence. Structure images are generated using Pseudoviewer [7]

When we compared the structures predicted by TAG_{RNA} and TAG_{RNA}Inf, we found out that in most cases where TAG_{RNA}Inf gave better predictions than TAG_{RNA}, the structures predicted by TAG_{RNA} had one bp stems in them. As mentioned earlier the default setting for the minimum stem size in TAG_{RNA}Inf grammars is two. TAG_{RNA} has an option to change the minimum stem size, but we tested all tools under their default settings. It is worthy to mention here, that the reported results for TAG_{RNA} take advantage of the identification phase performed by TAG_{RNA}Inf, and these results will endure a drop when we eliminate this phase in the second experiment.

In the second experiment, we included all 36 sequences when calculating folding sensitivity and specificity, ignoring the results of the identification phase. The numbers in Table 2 show a drop in the results, compared to Table 1, across all prediction tools, except for the specificity of ILM. If we look closer to the prediction results for the 5 rejected sequences, listed in table 3, we will observe the following: 2 out of the 5 rejected sequences (BSBV2 and BYDV-NY-RPV) give low sensitivity and specificity values across all prediction tools. Additionally, TRV-PSG and oligo-PK5 give low sensitivity on TAG_{RNA}Inf, This explains the general improvement achieved when these sequences are removed from the test set, specially for TAG_{RNA}Inf. Note that according to the framework we are presenting, identification must be performed prior to structure prediction. Thus, if a sequence was not identified to have a structure that belongs to the family of grammars under consideration, the structure predicted for this sequence by TAG_{RNA}Inf, if any, is considered to have a low confidence level.

In addition to the previous two experiments, we ran a third experiment in which we added two non-pseudoknot RNA grammars to the inferred six grammars. The two added grammars were for hair-pin RNA structures. The aim of this experiment was to test the effect of broadening the search space beyond the structures of the training set and compare it with the other approaches. We realize that the search space of the other tools is much broader. It is worth noting here that our approach is structure prediction through grammar learning. In this experiment, the two grammars were added, without going through the learning phase. Also, a threshold function was not inferred, and no identification was done before the structure prediction.

Table 4 includes the results for predicting the structure of the 36 input sequences with and without the added non-pseudoknot grammars. It also includes the results for pknotsRG in enforce pseudoknot mode and mfe mode. The mfe mode predicts the minimum free energy structure without trying to enforce a pseudoknot structure. We report sensitivity, specificity, and number of sequences whose predicted structure differed due to introducing the non-pseudoknot grammars for TAG_{RNA}Inf, or switching to mfe mode in the case of pknotsRG.

Table 3. Folding sensitivity and specificity for the five sequences rejected by the H-type pseudoknot grammars on TAG_{RNA}Inf

Sequence	ILM		pknotsRG(enf)		TAG _{RNA}		TAG _{RNA} Inf	
	Sen	Spec	Sen	Spec	Sen	Spec	Sen	Spec
TRV-PSG [4][32]	100	92.9	100	100	38.5	35.7	61.5	100
BSBV2 [4][15]	25.0	27.3	41.7	38.5	83	83	41.7	55.6
BYDV-NY-RPV[4]	0.0	0.0	22.0	22.0	0.0	0.0	0.0	0.0
Oligo-PK5 [4]	100	100	100	100	100	100	75.0	100
Azoacus_BH72(PK1) [34]	100	100	100	100	56.0	45.0	89.0	100

Table 4. Folding sensitivity and specificity for the whole 36 sequence test set on pknotsRG in both enforce mode (enf) and mfe mode and on TAG_{RNA}Inf using the inferred H-type pseudoknot grammars (PK) and with the added two hair-pin grammars (nonPK)

	Sensitivity	Specificity	Affected sequences n/36
pknotsRG (enf)	75.5	77.0	-
pknotsRG (mfe)	75.0	76.7	5
TAG _{RNA} Inf (PK)	79.5	85.3	-
TAG _{RNA} Inf (nonPK)	78.0	85.7	4

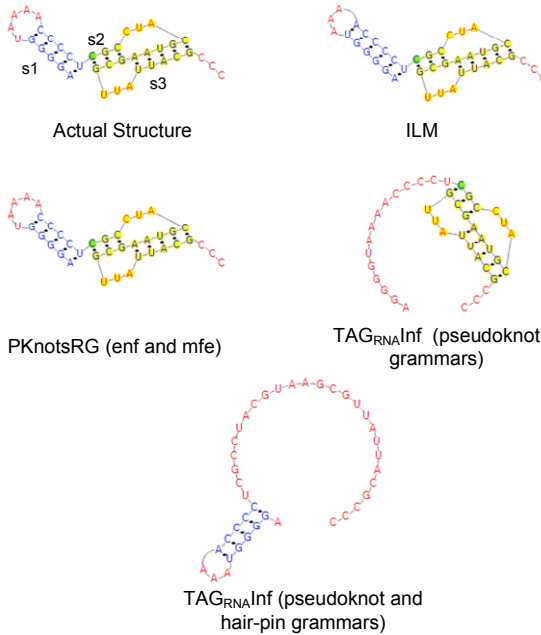


Fig. 5. Actual [32] and predicted structures for TRV-PSG RNA sequence. Structure images are generated using Pseudoviewer [7].

We notice more stability in the sensitivity and specificity of pknotsRG than TAG_{RNA}Inf. We found that 5 out of the 36 structures were affected in the case of pknotsRG and 4 in the case of TAG_{RNA}Inf. There was no overlap between the sequences affected for each approach. We also found that 3 out of the 4 affected sequences in the case of TAG_{RNA}Inf belong to the set of rejected (unidentified) sequences reported in Table 3. This once more suggests the benefit of using the identification before prediction idea as an indicator of the confidence level of the predicted structure.

To illustrate the advantage of this methodology further, we will present an example, TRV-PSG, which gave considerably worse sensitivity and specificity on TAG_{RNA}Inf, compared to some of the other approaches. The actual structure for

TRV-PSG [32], illustrated in Fig. 5, consists of a hair-pin concatenated to a H-type pseudoknot. PknotsRG gave perfect structure prediction of TRV_PSG in both enf and emf modes. ILM gave almost perfect structure prediction where it predicted one additional bp in the hair-pin stem (s1). TAG_{RNA}Inf with the H-type pseudoknot grammar set totally missed the hair-pin structure (s1). On the other hand, with the added two hair-pin grammars, TAG_{RNA}Inf predicted a hair-pin structure for TRV-PSG, and the pseudoknot (s2 and s3) was missed. Notice once more that the identification phase was able to recognize that the structure of TRV-PSG does not belong to the structures represented by the inferred set of grammars. TAG_{RNA}Inf could have been able to predict the correct structure for TRV-PSG if the training data set included a sequence that had a similar structure, a hair-pin concatenated to the pseudoknot.

5 Conclusion and Future Directions

In this paper we presented an RNA structure identification/prediction framework, TAG_{RNA}Inf capable of handling pseudoknot structures. The framework is a variant of our previous work [1]. In this framework, if a certain structure is identified in a sequence, a folding is computed for the sequence. Our experimental results show the advantage of the identification step to the folding performed by TAG_{RNA}Inf as well as other folding approaches.

In this paper and in [1], we discussed two problems related to RNA structure analysis, which are structure identification and structure prediction. In future work, we plan to address the problem of structural classification. Our preliminary results showed that the use of grammars alone would not be sufficient for classification. We plan to investigate the coupling of the grammatical methods with sequence based methods to do structural classification.

As mentioned earlier, the parser used within TAG_{RNA}Inf's identification/prediction phase is an implementation of the SLTAG/ESLTAG parsers described in [31]. These algorithms use a $(n+1)^4$ matrix and they can be described as metric-centric. Independent of the elementary trees in the given grammar, the algorithms step through each entry in this matrix to check if any trees can be placed in this entry. This means that even if no tree will ever be placed in a matrix entry, some work is done corresponding to this entry. In [21], we describe new parsing algorithms for SLTAG and ESLTAG which are tree-centric. These algorithms are expected to be more time efficient in practice. Additionally, since the matrix used by the parsers is usually sparse, we intend to use other data structures that will result in reducing the space requirements as well. We plan to provide comparative results to prove the vantage of the new algorithms in practice. Additionally, we will work on designing a parallelized version of these algorithms.

References

1. Al Seesi, S., Rajasekaran, S., Ammar, R.: Pseudoknot Identification through Learning TAG_{RNA}. In: Chetty, M., Ngom, A., Ahmad, S. (eds.) PRIB 2008. LNCS (LNBI), vol. 5265, pp. 132–143. Springer, Heidelberg (2008)
2. Akutsu, T.: Dynamic Programming Algorithms for RNA Secondary Structure Prediction with Pseudoknots. *Discrete Applied Mathematics* 104, 45–62 (2000)

3. Ambros, V., Bartel, B., Bartel, D.P., Burge, C.B., Carrington, J.C., Chen, X., Dreyfuss, G., Eddy, S.R., Griffiths-Jones, S., Marshall, M., Matzke, M., Ruvkun, G., Tuschl, T.: A Uniform System for microRNA Annotation. *RNA* 9(3), 277–279 (2003)
4. Batenburg, Dvan, F.H., Gulyaev, A.P., Pleij, C.W.A., Ng, J., Oliehoek, J.: Pseudobase: a Database with RNA Pseudoknots. *Nucl. Acids Res.* 28(1), 201–204 (2000)
5. Brazma, A., Jonassen, I., Vilo, J., Ukkonen, E.: Pattern Discovery in Biosequences. In: Honavar, V.G., Slutzki, G. (eds.) *ICGI 1998*. LNCS, vol. 1433, pp. 255–270. Springer, Heidelberg (1998)
6. Buratti, E., Dhir, A., Lewandowska, M.A., Baralle, F.E.: RNA Structure is a Key Regulatory Element in Pathological ATM and CFTR Pseudoxon Inclusion Events. *Nucl. Acids Res.* 35(13), 4369–4383 (2007)
7. Byun, Y., Han, K.: PseudoViewer: Web application and Web service for Visualizing RNA Pseudoknots and Secondary structures. *Nucl. Acids Res.* 34, W416–W422 (2006)
8. Chiu, D., Kolodziejczak, T.: Inferring consensus structure from nucleic acid sequences. *Comput. Appl. Biosci.* 7, 347–352 (1991)
9. Dirks, R.M., Pierce, N.A.: A Partition Function Algorithm for Nucleic Acid Secondary Structure Including Pseudoknots. *J. Comput. Chem* 24(13), 1664–1677 (2003)
10. Gilbert, W.: The RNA World. *Nature* 319, 618 (1986)
11. Griffiths-Jones, S., Moxon, S., Marshall, M., Khanna, A., Eddy, S.R., Bateman, A.: Rfam: Annotating Non-coding RNAs in Complete Genomes. *Nucl. Acids Res.* 33, D121–D124 (2005)
12. Gulko, B., Haussler, D.: Using multiple alignments and phylogenetic trees to detect RNA secondary structure. In: *Proc. Pac. Symp. Biocomput.* vol. 1, pp. 350–367
13. Hofacker, I.L., Fontana, W., Stadler, P.F., Bonhoeffer, S., Tacker, M., Schuster, P.: Fast Folding and Comparison of RNA Secondary Structures. *Monatshefte f. Chemie* 125, 167–188 (1994)
14. Joshi, A.K., Levy, L., Takahashi, M.: Tree Adjunct Grammars. *Journal of Computer and System Sciences* 10, 136–163 (1975)
15. Koenig, R., Commandeur, U., Loss, S., Beier, C., Kaufmann, A., Lesemann, D.-E.: Beet Soil-borne Virus RNA 2: Similarities and Dissimilarities to the Coat Protein Gene-carrying RNAs of other Furoviruses. *J. Jen. Virol.* 78, 469–477 (1997)
16. Laxminarayana, J.A., Nagaraja, G., Balaji, P.V.: Identification of Pseudoknots in RNA Secondary Structures: A Grammatical Inference Approach. In: Mukherjee, D.P., Pal, S. (eds.) *Proceedings of 5th International Conference on Advances in Pattern Recognition* (2003)
17. Lyngso, R., Pedersen, C.: RNA pseudoknot prediction in energy-based models. *J. Comput. Biol.* 7, 409–427 (2000)
18. Nussinov, R., Pieczenik, G., Griggs, J., Kleitman, D.: Algorithms for loop matchings. *SIAM J. Appl. Math.* 35, 68–82 (1978)
19. Paillart, J.C., Skripkin, E., Ehresmann, B., Ehresmann, C., Marquet, R.: In vitro Evidence for a Long Range Pseudoknot in the 5'-Untranslated and Matrix Coding regions of HIV-1 Genomic RNA. *J. Biol. Chem.* 277, 5995–6004 (2002)
20. Pedersen, J.S., Bejerano, G., Siepel, A., Rosenbloom, K., Lindblad-Toh, K., Lander, E.S., Kent, J., Miller, W., Haussler, D.: Identification and Classification of Conserved RNA Secondary Structures in the Human Genome. *Public Library of Science. Computational Biology* 2(4), e33 (2006)
21. Rajasekaran, S., Al Seesi, S., Ammar, R.: Improved Algorithms for Parsing ESLTAG: a grammatical model suitable for RNA pseudoknots. In: *International Symposium on Bioinformatics Research and Applications, ISBRA* (submitted, 2009)

22. Reeder, J., Giegerich, R.: Design, Implementation and Evaluation of a Practical Pseudoknot Folding Algorithm Based on Thermodynamics. *BMC Bioinformatics* 5, 104 (2004)
23. Rivas, E., Eddy, S.: A dynamic programming algorithm for RNA structure prediction including pseudoknots. *J. Mol. Biol.* 258, 2053–2068 (1999)
24. Robertson, M.P., Igel, H., Baertsch, R., Haussler, D., Ares, M., Scott Jr., W.G.: The Structure of a Rigorously Conserved RNA Element within the SARS Virus Genome. *Public Library of Science: Biology* 3(1), e5 (2004)
25. Ruan, J., Stormo, G.D., Zhang, W.: An Iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics* 20(1), 58–66 (2004)
26. Sakakibara, Y., Brown, M., Hughey, R., Mian, I.S., Sjolander, K., Underwood, R.C., Haussler, D.: Stochastic Context-Free Grammars for tRNA Modeling,” *Nucl. Acids Res.* 22, 5112–5120 (1994)
27. Sakakibara, Y.: Grammatical Inference in Bioinformatics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 1051–1062 (2005)
28. Searls, D.: The Linguistics of DNA. *Am. Scient* 80, 579–591 (1992)
29. Takakura, T., Asakawa, H., Seki, S., Kobayashi, S.: Efficient Tree Grammar Modeling of RNA Secondary Structures from Alignment Data. In: *Proceedings of posters of RECOMB 2005*, pp. 339–340 (2005)
30. Tanaka, Y., Hori, T., Tagaya, M., Sakamoto, T., Kurihara, Y., Katahira, M., Uesugi, S.: Imino Proton NMR Analysis of HDV Ribozymes: Nested Double Pseudoknot Structure and Mg²⁺ Ion-Binding Site Close to the Catalytic Core in Solution. *Nucl. Acids Res.* 30, 766–774 (2002)
31. Uemura, Y., Hasegawa, A., Kobayashi, S., Yokomori, T.: Tree Adjoining Grammars for RNA Structure Prediction. *Theoretical Computer Science* 210(2), 277–303 (1999)
32. van Belkum, A., Cornelissen, B., Linthorst, H., Bol, J., Pley, C., Bosch, L.: tRNA-like Properties of Tobacco Rattle Virus RNA. *Nucl. Acids Res.* 15(7), 2837–2850 (1987)
33. Vijay-Shanker, K., Joshi, A.K.: Some Computational Properties of Tree Adjoining Grammars. In: *23 rd Meeting of the Association for Computational Linguistics*, pp. 82–93 (1985)
34. Williams, K.P.: The tmRNA Website: Invasion by an Intron. *Nucl. Acids Res.* 30(1), 179–182 (2002)
35. Zuker, M., Stiegler, P.: Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res.* 9, 133–148 (1981)

Comparing Bacterial Genomes by Searching Their Common Intervals

Sébastien Angibaud, Damien Eveillard, Guillaume Fertin, and Irena Rusu

Computational Biology group (ComBi)

Laboratoire d'Informatique de Nantes-Atlantique (LINA), UMR CNRS 6241,
Université de Nantes, 2 rue de la Houssinière, 44322 Nantes Cedex 3, France

Tel.: +33 2.51.12.58.17; Fax: +33 2.51.12.58.97

`sebastien.angibaud@univ-nantes.fr`, `damien.eveillard@univ-nantes.fr`,
`guillaume.fertin@univ-nantes.fr`, `irena.rusu@univ-nantes.fr`

Abstract. Comparing bacterial genomes implies the use of a dedicated measure. It relies on comparing circular genomes based on a set of conserved genes. Following this assumption, the common interval appears to be a good candidate. For evidences, we propose herein an approach to compute the common intervals between two circular genomes that takes into account duplications. Its application on a concrete case, comparing *E. coli* and *V. cholerae*, is accurate. It indeed emphasizes sets of conserved genes that present high impacts on bacterial functions.

Keywords: comparative genomic, circular genome, common interval.

1 Introduction

The bacterial world is ubiquitous and shows a great diversity [21]. Interestingly, it is responsible for the major key biological processes, but it remains far from being well-understood. For a while, understanding the bacterial diversity was considered as a pointless area because of the complexity and the versatility of this bacterial world, and because of its dynamic nature. Recent efforts in high-throughput methods for analyzing bacterial genomes, confirm its dynamic but emphasize, as well, a more comprehensive picture of the structure of genomes [12]. It gives various insights for investigating the bacterial complexity [6]. In particular, Doolittle [7] suggests sets of genes as convenient descriptors for comparing two bacterial species. Following this assumption, comparing two bacterial genomes implies (i) comparing their genes for finding orthologs and (ii) finding their invariants, which indicates the genes that are conserved in both species. Furthermore, it highlights common biological properties shared by the compared bacteria.

Previous points raises several computational questions. The first one deals with the inherent complexity of comparing one gene of a bacterial genome with genes from another genome. In [4], Berglund et al. tackled this problem by using a clustering technique as implemented in the *InParanoid* software. The second one concerns the choice of an appropriate measure to compare genome structures.

Based on a correct mapping of orthologs between species, various approaches propose a theoretical framework to compute a measure between genomes based on their structure similarities [5]. They particularly highlight the sets of genes that are conserved among the genomes.

When applied on prokaryotic world and its particular features, comparing genomes implies the use of a dedicated protocol. By nature, bacterial genomes are circular, which must be taken into account by a dedicated measure. In addition, experimental investigations show evidences of high rates of gene duplications in bacteria. Based on these assumptions, this paper depicts a *in silico* protocol for the bacterial genome comparison. We propose (i) to find bacterial homologies by using the `InParanoid` software. We then have to put forward (ii) a dedicated measure. It relies on comparing circular genomes based on a set of conserved genes (following the assumption detailed in [7] and mentioned above). In this purpose, we propose an adaptation of the *common interval* [20] with a special emphasis on the circularity and duplications.

The paper is organized as follows. We briefly give notations and definitions in Section 2. We show in Section 3 how to compute the number of common intervals between two circular genomes that takes into account duplications. Based on these theoretical features, we are able to define a complete approach to compare bacterial genomes. In Section 4, we propose its application on a concrete case: comparing *Escherichia coli* and *Vibrio cholerae*. These well-known γ -Proteobacteria appear as an appropriate benchmark for testing the measure (quantitatively and qualitatively). Beyond the accuracy of the comparison, the measure emphasizes sets of genes that are conserved on genomes. These genes show particular functional properties and belong to operons, which reinforces the biological relevance of the common interval measure in comparative genomics.

2 Preliminaries

The following section gives some notations and definitions used in the paper.

Notations. Let \mathcal{F} be a set of *genes*, where each gene is represented by an integer. A circular genome G is represented by an *ordered* sequence of signed elements (*signed genes*) from \mathcal{F} , where we consider that the first gene and the last one are adjacent. Denote η_G the size of genome G . Let $G[p]$, $0 \leq p \leq \eta_G - 1$, be the signed gene that occurs at position p on a genome G . For any signed gene g , let \bar{g} be the signed gene having the opposite sign. Let $occ_G(g)$, $g \in \mathcal{F}$, be the number of occurrences of the gene g in G . Given a genome G without duplicates (i.e. without signed genes having the same absolute value) and two signed genes a , b , let $G[a, b]$ be the set of unsigned genes located between genes a and b in G according to the circular order ($G[0] G[1] \dots G[\eta_G - 1] G[0] \dots$) on G . We also note $[a, b]_G$ the substring (i.e. the sequence of consecutive elements) of G starting by a and finishing by b .

For example, consider the set $\mathcal{F} = \{1, 2, 3, 4, 5, 6\}$ and the circular genome $G = +3 - 2 + 6 + 4 - 1 + 5$ without duplicates. Then, $G[4] = -1$ and $\overline{G[4]} = +1$, $G[-1, +3] = \{1, 3, 5\}$ and $[-1, +3]_G = -1 + 5 + 3$.

Common intervals. We define here the measure used in the paper. Let G_1, G_2 be two circular genomes without duplicates and with a similar gene content. A *common interval* [20] of (G_1, G_2) is a substring of G_1 such that G_2 contains a permutation of this substring (not taking signs into account). For example, consider $G_1 = +1 + 2 + 3 + 4 + 5$ and $G_2 = +2 - 4 + 3 + 5 + 1$. Then, substring $[+3, +5]_{G_1}$ is a common interval of (G_1, G_2) . Note that the original definition of a common interval of [20] is extended here to consider circular sequences. Therefore, substring $[+5, +1]_{G_1}$ is also a common interval.

Given a set \mathcal{I} of common intervals, a common interval $I \in \mathcal{I}$ is called *maximal*, if there is no common interval of \mathcal{I} that strictly contains it. We call a *straight* interval, a maximal common interval for which the order and the signs of its genes are identical in G_1 and G_2 . On the contrary, the maximal interval is called *reverse* if the order and the signs of genes are reversed in G_1 with respect to G_2 . Finally, we call an *unstructured* interval, a maximal common interval that is neither straight nor reverse.

Genomes with duplicates. When genomes contain duplicates, we cannot directly compute the number of common intervals, because this measure is defined on permutations. A natural solution consists in i) finding a one-to-one correspondence (i.e. a matching) between signed genes of G_1 and G_2 , ii) using this correspondence to rename genes of G_1 and G_2 , and iii) deleting the unmatched signed genes in order to obtain two genomes G'_1 and G'_2 such that G'_2 is a *permutation* of G'_1 . Computing the measure becomes thus possible. Our study proposes to focus on two matching models: the *exemplar* model [15] and the *maximum matching* model [18].

- *Exemplar model:* for each gene g , we keep in the matching only one occurrence of g in G_1 and in G_2 .
- *Maximum matching model:* this model keeps the maximum number of signed genes in both genomes. In particular, we look for a one-to-one correspondence between signed genes of G_1 and G_2 that matches, for each gene g , exactly $\min(\text{occ}_{G_1}(g), \text{occ}_{G_2}(g))$ occurrences.

For a given model, among all possible matchings, we look for one that optimizes the number of common intervals. However, given two genomes G_1 and G_2 , the problem that consists in finding an exemplar (resp. a maximum matching) of (G_1, G_2) such that the number of common intervals is maximized, has been proved to be **APX-Hard** [1]. This complexity holds even when G_1 does not contain duplicates and each gene appears at most twice in G_2 .

3 Maximizing the Number of Common Intervals between Two Circular Genomes

In [2], authors proposed three methods to compare two genomes that are modeled as linear sequences of genes. The first one is an exact algorithm based on transforming an optimization problem into a 0–1 linear program [16]. The second

one is a heuristic based on the notion of Longest Common Substring (*LCS*) and the third method is an hybrid method that combines both previous approaches. An approach dedicated to the bacterial genome analysis implies to take into account the natural circularity of the input genomes. For that, we propose to show herein how to extend previous works.

3.1 Exact Approach

The principle of this approach is based on (1) transforming an optimization problem into a 0–1 linear program [16] and (2) run this program on a powerful solver (e.g. MiniSat+ [8]) in order to obtain optimal solutions. Given two circular genomes G_1 and G_2 , we detail the above transformation for both *exemplar* and *maximum matching* models.

Variables. We define two types of variables: the *match variables* and the *interval variables*. One *match variable* is defined for each possible pair of signed genes that can be matched together. Any such variable will be set to 1 if the pair of corresponding signed genes is matched, and 0 otherwise. Thus, we define the set of match variables as follows:

$$X = \{x_j^i : 0 \leq i < \eta_{G_1} \wedge 0 \leq j < \eta_{G_2} \wedge |G_1[i]| = |G_2[j]|\}$$

$$\forall x_j^i \in X, x_j^i \in \{0, 1\}$$

The *interval variables* correspond to the possible common intervals. Note that the whole genome G_1 is necessarily a common interval. Hence, we do not consider these intervals in order to reduce the linear program generated. These variables are defined as follows:

$$C = \{c_{j,m}^{i,n} : 0 \leq i < \eta_{G_1} \wedge 0 \leq j < \eta_{G_2} \wedge 0 \leq n < \eta_{G_1} - 1 \wedge 0 \leq m < \eta_{G_2} - 1\}$$

$$\forall c_{j,m}^{i,n} \in C, c_{j,m}^{i,n} \in \{0, 1\}$$

A variable $c_{j,m}^{i,n} \in C$ corresponds to the common interval that begins at the position i on G_1 and that contains $n + 1$ signed genes. Its corresponding interval on G_2 begins at the position j and contains $m + 1$ signed genes.

Objective function. We want to maximize the number of common intervals between G_1 and G_2 . For that, we define the objective function as the sum of the interval variables:

$$\text{maximize } \sum_{c_{j,m}^{i,n} \in C} c_{j,m}^{i,n}$$

Constraints. We define several constraints to insure that the assigned variables correspond to a valid matching, according to the considered model. For that, we first define two constraints to verify that each signed gene cannot be matched to more than one signed gene on the other genome:

$$(C1.a) \forall i, 0 \leq i < \eta_{G_1}, \sum_{\substack{0 \leq j < \eta_{G_2} \\ |G_1[i]| = |G_2[j]|}} x_j^i \leq 1$$

$$(C1.b) \forall i, 0 \leq j < \eta_{G_2}, \sum_{\substack{0 \leq i < \eta_{G_1} \\ |G_1[i]| = |G_2[j]|}} x_j^i \leq 1$$

Then, for each gene, we count the number of signed genes that are matched in order to respect the definition of the model. For the *maximum matching* model, we must have:

$$(C2) \forall g \in \mathcal{F}, \sum_{\substack{0 \leq i < \eta_{G_1} \\ |G_1[i]| = g}} \sum_{\substack{0 \leq j < \eta_{G_2} \\ |G_2[j]| = g}} x_j^i = \min\{occ_{G_1}(g), occ_{G_2}(g)\}$$

For the *exemplar* model, we must have:

$$(C2') \forall g \in \mathcal{F}, \sum_{\substack{0 \leq i < \eta_{G_1} \\ |G_1[i]| = g}} \sum_{\substack{0 \leq j < \eta_{G_2} \\ |G_2[j]| = g}} x_j^i = \min\{1, \min\{occ_{G_1}(g), occ_{G_2}(g)\}\}$$

In order to the interval variable validity, we introduce a new notation. For any $q \in \{1, 2\}$, we let $f_q(i, p) \equiv (i+p) \bmod \eta_{G_q}$. This notation is necessary for taking into account the common intervals that contain both $G_q[0]$ and $G_q[\eta_{G_q} - 1]$.

First, we must make sure that each extremity of a common interval is matched with the two following constraints:

$$(C3.a) \forall c_{j,m}^{i,n} \in C, 2c_{j,m}^{i,n} - \sum_{\substack{0 \leq p \leq m \\ |G_1[i]| = |G_2[f_2(j,p)]}} x_{f_2(j,p)}^i - \sum_{\substack{0 \leq p \leq m \\ |G_1[f_1(i,n)]| = |G_2[f_2(j,p)]}} x_{f_2(j,p)}^{f_1(i,n)} \leq 0$$

$$(C3.b) \forall c_{j,m}^{i,n} \in C, 2c_{j,m}^{i,n} - \sum_{\substack{0 \leq p \leq n \\ |G_1[f_1(i,p)]| = |G_2[j]|}} x_j^{f_1(i,p)} - \sum_{\substack{0 \leq p \leq n \\ |G_1[f_1(i,p)]| = |G_2[f_2(j,m)]}} x_{f_2(j,m)}^{f_1(i,p)} \leq 0$$

Then, we define constraints to insure that each signed gene of G_1 inside a common interval I is correctly matched. For that, we consider two cases. If the corresponding interval of I on G_2 does not contain both $G_2[0]$ and $G_2[\eta_{G_2} - 1]$, we write:

$$(C4.a) \forall c_{j,m}^{i,n} \in C, j + m < \eta_{G_2}, \forall 1 \leq p < n, \forall 0 \leq r < j, \\ |G_1[f_1(i, p)]| = |G_2[r]|, c_{j,m}^{i,n} + x_r^{f_1(i,p)} \leq 1$$

$$(C4.b) \forall c_{j,m}^{i,n} \in C, j + m < \eta_{G_2}, \forall 1 \leq p < n, \forall j + m < r \leq \eta_{G_2}, \\ |G_1[f_1(i, p)]| = |G_2[r]|, c_{j,m}^{i,n} + x_r^{f_1(i,p)} \leq 1$$

Else, if the corresponding interval of I on G_2 contains both $G_2[0]$ and $G_2[\eta_{G_2} - 1]$, we write:

$$(C4.c) \forall c_{j,m}^{i,n} \in C, j + m \geq \eta_{G_2}, \forall 1 \leq p < n, \forall f_2(j, m) < r < j,$$

$$|G_1[f_1(i, p)]| = |G_2[r]|, c_{j,m}^{i,n} + x_r^{f_1(i,p)} \leq 1$$

Finally, we also define the three symmetric constraints of (C4.a), (C4.b) and (C4.c) to consider each signed gene in G_2 .

3.2 Non-exact Approaches

IILCS heuristic. The *IILCS* heuristic proposed in [2] is a greedy algorithm based on the notion of Longest Common Substring (*LCS*). The program matches genes of an *LCS* of the two genomes, up to a complete reversal, and iterates this process until no gene can be matched (see [2] for more details). We easily modify this algorithm in order to compare circular genomes by identifying the *LCS* that may overlap the end and the beginning of the genomes.

Hybrid method. This approach uses both previous methods. First, a partial matching is obtained by running *IILCS* until the size of any *LCS* is smaller than a given parameter k , chosen by the user. Then, a 0–1 linear program is generated in order to match the remaining unmatched genes. Since both previous methods have been extended for taking into account genome circularity, the hybrid method is hence adapted to circular genomes.

4 Comparing Bacterial Genomes: A Practical Application

Based on the previous theoretical framework, we are now able to propose a practical comparison of proteobacterias. Here, we first define the protocol used and, secondly, put forward a precise analysis of biological results obtained.

4.1 Protocol

The different steps of our comprehensive approach might be described as follow (see also Figure 1 for illustration).

Step 1. Input data. One selects on the NCBI website two genomes G_1 and G_2 and transforms the corresponding data into two files in the FASTA format, one for each genome. These files contain the list of genes, each of which is described by a label followed by its protein sequence.

Steps 2 and 3. Homologies detection. We call the *InParanoid* software [13], which clusters orthologs and inparalogs genes based on a Blastp (step 2). The output file contains the clusters of the homologous genes. According to these sets, we tag these homologous genes with a similar label. It hence builds two intermediate genomes G'_1 and G'_2 (step 3).

Steps 4, 5 and 6. Measure computation. We use one of the approaches described in Section 3 to obtain a matching between the two genomes (step 4). In this purpose, two parameters must be specified:

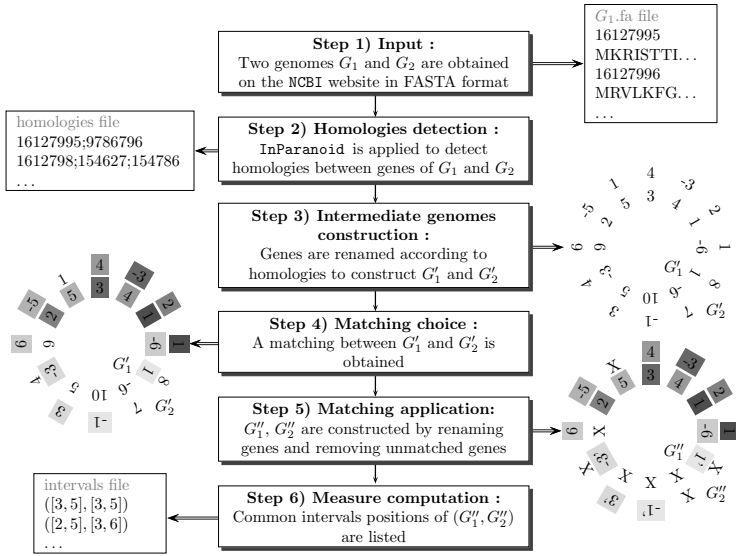


Fig. 1. Step by step description of the bacterial genome analysis

- *The model:* exemplar or maximum matching model (see Section 2).
- *The method:* the exact method based on a pseudo-boolean programming (PSB), the IILCS heuristic or the hybrid method with a parameter k bounding the size of the LCS (see Section 3).

We generate the list of gene pairs that are matched. Thus, we rename the genes according to this matching and remove the unmatched genes to construct two genomes G''_1 and G''_2 without duplications and with the same gene content (step 5). We then compute the common intervals of (G''_1, G''_2) (step 6).

Filtering relevant common intervals. Among the set \mathcal{I} of common intervals we have obtained, we select the relevant common intervals as follows:

1. We remove common intervals that contain all matched signed genes (the whole genome being a trivial common interval).
2. Since genomes are circular, for each common interval $I \in \mathcal{I}$, there exists a unique common interval $I' \in \mathcal{I}$, such that I' contains all matched signed genes that do not belong to I . The pair (I, I') will be called a *complementary pair*. Given any complementary pair (I, I') , we assume that the smallest interval $I_s \in \{I, I'\}$ (i.e., the one that contains less signed genes) is the most biologically informative, and thus we choose to keep only I_s in our set.
3. However, an interval that contains only one gene carries no biological information. We thus remove such intervals from our set.

4.2 Practical Application

We apply our method on a benchmark composed of proteobacteria’s genomes (*Escherichia coli* and *Vibrio cholerae*). Table 1 lists the considered set of genomes and Table 2 shows quantitative details concerning common intervals under the maximum matching model. Note that the computational time of InParanoid is much longer than the one needed to obtain the matching. However, for the longest sequences (*NC_000913* vs *NC_002505* or *NC_000913* vs *NC_009457*), only the *IILCS* heuristic gives results in an acceptable amount of time. Note also the distribution of straight, reverse and unstructured intervals (see Table 2). On average, 41% of maximal common intervals are straight, 49% are reverse and 10% are unstructured.

These quantitative results are computationnaly relevant. We therefore propose to go further by investigating their qualitative properties. In particular, we focus on the comparison of the chromosome pair composed by *Escherichia coli* (*NC_000913*) and *Vibrio cholerae* (*NC_009457*), for which the quantitative results are given in Table 2. We consider their comparison as an appropriate benchmark for testing the biological properties highlighted by the common interval measure. As shown in Figure 2, it emphasizes three kinds of common intervals:

Straight intervals. They show a perfectly conserved gene arrangements like in Figure 2(a). Based on experimental knowledges refered in EcoCyc [11], *CcmA*, *CcmB*, *CcmC*, *CcmD*, *CcmE*, *CcmF*, *CcmG* and *CcmH* are the genes that encode for proteins that are sub-units of the cytochrome C complex. These genes belong to the same operon promoted by *ccmAp* [17]. In particular, experimental studies show that mutants of one of these genes are deficient in the ability to produce c-type cytochromes [19]. Their presence into a unique straight common interval, emphasizes the fact that these genes are conserved by their DNA sequence (i.e. homology shown using InParanoid), but also by their arrangement on the bacterial genomes (i.e. determination of the common interval). This result confirms the interest to find highly functionnal genes into a single common interval.

Reverse intervals. They show sets of genes that presents a conservation between two genomes in a reverse order, like the one in Figure 2(b). This particular interval depicts the pilus gene clusters that encode for extracellular pilus structures. They are common among bacteria and have been involved in several colonization functions, like those involved in the intestinal mucosa colonization. For clinical

Table 1. Set of genomes analyzed

NCBI label	Name
NC_000913	<i>Escherichia coli</i> K12
NC_002505	<i>Vibrio cholerae</i> 01 biovar eltor str. N16961 chromosome I
NC_002506	<i>Vibrio cholerae</i> 01 biovar eltor str. N16961 chromosome II
NC_009456	<i>Vibrio cholerae</i> 0395 chromosome I
NC_009457	<i>Vibrio cholerae</i> 0395 chromosome II

Table 2. Relevant common intervals obtained under the maximum matching model by comparing *Escherichia coli* (NCBI label NC_000913) with four *Vibrio cholerae* chromosomes (NC_002505, NC_002506, NC_009456 and NC_009457)

genome G_2	genome size		method	computational time		relevant common intervals cardinality				
	<i>E. coli</i>	G_2		Inparanoid (s)	matching (s)	number	maximal	straight	reverse	unstructured
NC_002505	4243	2742	<i>IILCS</i>	1144	62	3710	275	117(43%)	134(48%)	24(9%)
NC_002506	4243	1093	<i>PSB</i>	638	23	123	50	18(36%)	23(46%)	9(18%)
NC_009456	4243	1133	<i>PSB</i>	651	22	132	55	17(31%)	27(49%)	11(20%)
NC_009457	4243	2742	<i>IILCS</i>	1199	59	3602	278	114(41%)	141(51%)	23(8%)

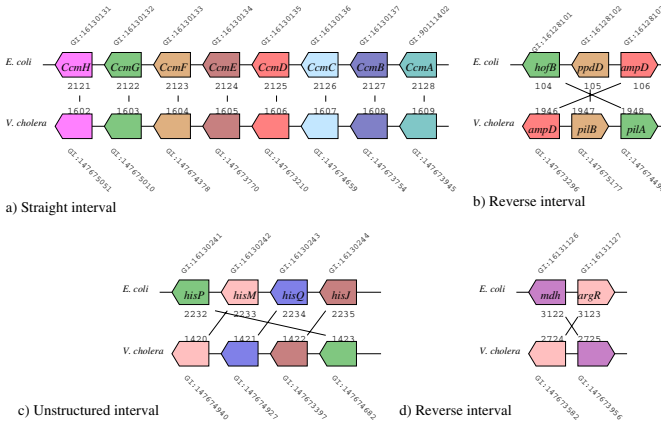


Fig. 2. Illustration of the three kinds of common intervals observed between *E. coli* (NC_000913) and *V. cholerae* (NC_009457). Each representation indicates the gene indexes (e.g., GI16130131), the genes position (e.g., 2121), the available labels of genes (e.g., *CcmH*), the gene orientation.

motivations, the homology of these genes is already shown between *V. cholerae* and *Vibrio fischeri* [14]. In [9], Fuller and Mekalanos depict the organization of the pilus assembly operon with 7 genes. The reverse common interval shown in Figure 2 b) suggests a conservation of 3 genes only, which is not fully accurate with the experimental assumption. Nevertheless, note that EcoCyc [10] not shows neither high-quality evidences of a common transcriptionnal unit for the seven genes, which not invalidates our method.

Unstructured intervals. They represent common intervals which are neither straight nor reverse. Figure 2 c) illustrates this case. Like in a), this particu-

lar interval also describes the arrangement of four genes that belong to the same operon: *hisJp* operon. It is activated by Hns and repressed by ArgR. It products a subunit of an ABC Transporter. Note again that *ArgR* that encodes ArgR belongs, with *mdh*, to a single reverse common interval (see Figure 2 d)). *mdh* produces Mdh that is a subunit of a malate dehydrogenase. It interacts in several biological processes (carbohydrate metabolism, gluconeogenesis, glycolysis, tricarboxylic acid cycle, tricarboxylic acid cycle intermediate metabolism, malate metabolism, fermentation, anaerobic respiration, glyoxylate catabolism, carbohydrate catabolism). Again, these results confirm that common intervals might be associated with operons, but moreover, like in this case, with a set of genes that controls operons. Assuming that a common interval emphasizes a biological function conservation over compared genomes, regulatory function appears as important as other metabolic functions. Like in this case, common interval shows a specific regulatory unit that controls distinct biological processes via distinct operons. It is particularly interesting for studying genomes and their comparisons from a functional viewpoint.

A close look at these qualitative results highlights that common intervals (either straight, reverse or unstructured) may indicate the presence of functional components within bacterial genomes. Such an information is particularly valuable for investigating the bacterial functional diversity, in particular in an environmental context for which the genomic data remain the corner stone for a better understanding.

5 Extension of the Method to Conserved Intervals

Apart of the common intervals, Bergeron and Stoye introduced in [3] the notion of *conserved intervals*. Given two circular genomes G_1 and G_2 with same gene content and without duplication, consider two signed genes a and b of G_1 , with possibly $a = b$. Substring $[a, b]_{G_1}$ is called a *conserved interval* of (G_1, G_2) if $[a, b]_{G_1}$ is a common interval and if it satisfies one of the two following properties: either a and b appear in G_2 and $G_1[a, b] = G_2[a, b]$; or \bar{a} and \bar{b} appear in G_2 and $G_1[a, b] = G_2[\bar{b}, \bar{a}]$. For example, if $G_1 = +1 + 2 + 3 + 4 + 5$ and $G_2 = -5 - 4 - 2 + 3 - 1$, substrings $[+4, +5]_{G_1}$ and $[+4, +1]_{G_1}$ are two conserved intervals of (G_1, G_2) .

Our previous approach (see Section 3) can be adapted to compute the conserved intervals instead of the common intervals. Concerning the exact approach, since a conserved interval is also a common interval with restrictions on its extremities, an additional filter on interval variables suffices to consider the number of conserved intervals. In that case, we change the set C of variables such that

$$C = \{c_{j,m}^{i,n} : 0 \leq i < \eta_{G_1} \wedge 0 \leq j < \eta_{G_2} \wedge 0 \leq n < \eta_{G_1} - 1 \wedge 0 \leq m < \eta_{G_2} - 1\} \wedge \\ ((G_1[i] = G_2[j] \wedge G_1[f_1(i, n)] = G_2[f_2(j, m)]) \vee \\ (G_1[i] = G_2[f_2(j, m)] \wedge G_1[f_1(i, n)] = G_2[j]))$$

For *IILCS* and hence for the hybrid method, the process remains the same in the greedy phase. However, the intervals computation is modified to count only conserved intervals, which does not affect the complexity of the algorithm.

6 Conclusion

In this paper, we have presented a comprehensive method to find sets of genes that are conserved between two circular genomes. We present for that an approach to compute or approximate the number of common intervals between two circular genomes. Each resulting interval can be considered as a set of genes that is conserved between the two genomes during the evolution process. Our method was tested on *Escherichia coli* and four chromosomes of *Vibrio cholerae*. From these experimentations, the results strongly suggest that common intervals is a measure that provide useful information on bacterial genomes and help the user to focus on specific sets of genes that possess fonctionnal and regulatory properties. It confirms the interest of the common interval measure for giving more fonctionnal insights in comparative genomics studies. A thorough biological analysis of each maximum common interval, along with tests on larger benchmarks, are planned in the very next future.

Acknowledgment. the authors thank Dr. Richard A. Long for the fruitful discussions that initiated this work.

References

1. Angibaud, S., Fertin, G., Rusu, I.: On the approximability of comparing genomes with duplicates. In: Nakano, S.-i., Rahman, M. S. (eds.) WALCOM 2008. LNCS, vol. 4921, pp. 34–45. Springer, Heidelberg (2008)
2. Angibaud, S., Fertin, G., Rusu, I., Vialette, S.: A pseudo-boolean general framework for computing rearrangement distances between genomes with duplicates. *Journal of Computational Biology* 14(4), 379–393 (2007)
3. Bergeron, A., Stoye, J.: On the similarity of sets of permutations and its applications to genome comparison. In: Warnow, T.J., Zhu, B. (eds.) COCOON 2003. LNCS, vol. 2697, pp. 68–79. Springer, Heidelberg (2003)
4. Berglund, A., Sjölund, E., Ostlund, G., Sonnhammer, E.L.L.: Inparanoid 6: eukaryotic ortholog clusters with inparalogs. *Nucleic Acids Res.* 36(Database issue), D263–D266 (2008)
5. Blin, G., Chauve, C., Fertin, G., Rizzi, R., Vialette, S.: Comparing genomes with duplications: A computational complexity point of view. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 4(4), 523–534 (2007)
6. Curtis, T.P., Sloan, W.T.: Prokaryotic diversity and its limits: microbial community structure in nature and implications for microbial ecology. *Curr. Opin. Microbiol.* 7(3), 221–226 (2004)
7. Doolittle, W.F.: Phylogenetic classification and the universal tree. *Science* 284(5423), 2124–2129 (1999)
8. Eén, N., Sörensson, N.: Translating pseudo-boolean constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation* 2, 1–26 (2006)
9. Fullner, K.J., Mekalanos, J.J.: Genetic characterization of a new type IV-a pilus gene cluster found in both classical and El Tor biotypes of *Vibrio cholerae*. *Infect. Immun.* 67(3), 1393–1404 (1999)

10. Karp, P.D., Keseler, I.M., Shearer, A., Latendresse, M., Krummenacker, M., Paley, S.M., Paulsen, I., Collado-Vides, J., Gama-Castro, S., Peralta-Gil, M., Santos-Zavaleta, A., Peñaloza-Spínola, M.I., Bonavides-Martinez, C., Ingraham, J.: Multidimensional annotation of the *Escherichia coli* K-12 genome. *Nucleic Acids Res.* 35(22), 7577–7590 (2007)
11. Keseler, I.M., Collado-Vides, J., Gama-Castro, S., Ingraham, J., Paley, S., Paulsen, I.T., Peralta-Gil, M., Karp, P.D.: Ecocyc: a comprehensive database resource for *Escherichia coli*. *Nucleic Acids Res.* 33(Database issue), D334–D337 (2005)
12. Ochman, H., Davalos, L.M.: The nature and dynamics of bacterial genomes. *Science* 311(5768), 1730–1733 (2006)
13. Remm, M., Strom, C.E., Sonnhammer, E.L.: Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *J. Molecular Biology* 314, 1041–1052 (2001)
14. Ruby, E.G., Urbanowski, M., Campbell, J., Dunn, A., Faini, M., Gunsalus, R., Lostroh, P., Lupp, C., McCann, J., Millikan, D., Schaefer, A., Stabb, E., Stevens, A., Visick, K., Whistler, C., Greenberg, E.P.: Complete genome sequence of *Vibrio fischeri*: a symbiotic bacterium with pathogenic congeners. *Proc. Natl. Acad. Sci. USA* 102(8), 3004–3009 (2005)
15. Sankoff, D.: Genome rearrangement with gene families. *Bioinformatics* 15(11), 909–917 (1999)
16. Schrijver, A.: *Theory of Linear and Integer Programming*. John Wiley and Sons, Chichester (1998)
17. Tanapongpipat, S., Reid, E., Cole, J.A., Crooke, H.: Transcriptional control and essential roles of the *Escherichia coli* ccm gene products in formate-dependent nitrite reduction and cytochrome C synthesis. *Biochem J.* 334(pt. 2), 355–365 (1998)
18. Tang, J., Moret, B.M.E.: Phylogenetic reconstruction from gene-rearrangement data with unequal gene content. In: Dehne, F., Sack, J.-R., Smid, M. (eds.) WADS 2003. LNCS, vol. 2748, pp. 37–46. Springer, Heidelberg (2003)
19. Thöny-Meyer, L.: Haem-polypeptide interactions during cytochrome C maturation. *Biochim. Biophys. Acta.* 1459(2-3), 316–324 (2000)
20. Uno, T., Yagiura, M.: Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica* 26(2), 290–309 (2000)
21. Whitman, W.B., Coleman, D.C., Wiebe, W.J.: Prokaryotes: the unseen majority. *Proc. Natl. Acad. Sci. USA* 95(12), 6578–6583 (1998)

Generalized Binary Tanglegrams: Algorithms and Applications

Mukul S. Bansal, Wen-Chieh Chang, Oliver Eulenstein, and David Fernández-Baca

Department of Computer Science, Iowa State University, Ames, IA - 50011, USA
{bansal, wcchang, oeulens, fernande}@cs.iastate.edu

Abstract. Several applications require the joint display of two phylogenetic trees whose leaves are matched by inter-tree edges. This issue arises, for example, when comparing gene trees and species trees or when studying the co-speciation of hosts and parasites. The *tanglegram layout* problem seeks to produce a layout of the two trees that minimizes the number of crossings between the inter-tree edges. This problem is well-studied for the case when the mappings between the leaves of the two trees is one-to-one. However, in typical biological applications, this mapping is seldom one-to-one. In this work we (i) define a generalization of the tanglegram layout problem, called the *Generalized Tanglegram Layout (GTL)* problem, which allows for arbitrary interconnections between the leaves of the two trees, (ii) provide efficient algorithms for the case when the layout of one tree is fixed, (iii) discuss the fixed parameter tractability and approximability of the GTL problem, (iv) formulate heuristic solutions for the GTL problem, and (v) evaluate our algorithms experimentally.

1 Introduction

The simultaneous examination of a species phylogeny and a gene phylogeny can offer biologists insights into evolutionary processes — such as gene duplication and loss, lateral gene transfer, and deep coalescence — that the inspection of either tree alone cannot provide [6, 13, 9]. The interaction between a gene tree and a species tree is customarily represented by a two-dimensional layout where the leaves of each tree are drawn on separate parallel lines, and where a straight line, called an *inter-tree edge*, connects each leaf (i.e., gene) in the gene tree with the leaf in the species tree (i.e., species) from which the gene was sampled. Each leaf in the species tree may in fact share inter-tree edges with multiple leaves in the gene tree, because a species may have several associated genes. The number of crossings between the inter-tree edges depends on the layout of the trees. A layout with many crossings can be nearly impossible to analyze. In extreme cases, the number of crossings in one drawing can be quadratic in the number of inter-tree edges, while a redrawing of the trees eliminates all crossings. The *tanglegram layout (TL) problem* is to find a layout of the two trees that minimizes the number of crossings. As illustrated in Fig. 1, the layout corresponding to an optimum solution to the TL problem can be a dramatic improvement over an unoptimized layout.

The TL problem has been studied by other researchers (see below); however, previous work has only dealt with the case where the mapping between the leaves of the two

¹ Throughout this work, all trees are assumed to be binary.

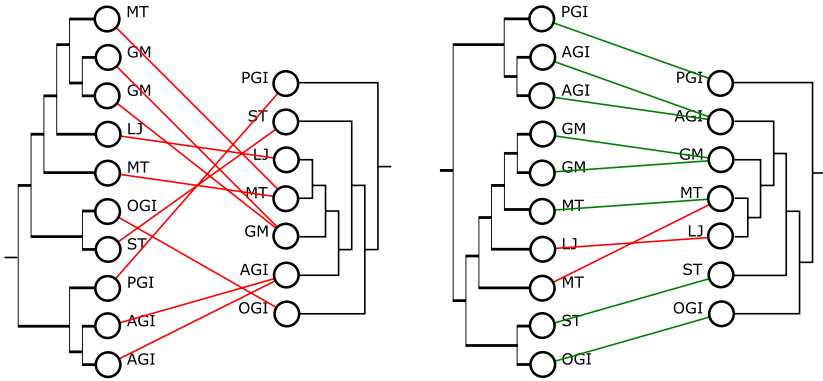


Fig. 1. Two layouts of a gene tree and a species tree, displaying the interactions between their leaves. An arbitrary layout is shown on the left; on the right is the optimized layout that results from solving the TL problem. Data is from [12].

trees is one-to-one. This special case does not handle the more general, and extremely useful, case where the mapping is many-to-many. Here, we define and study the general version of the TL problem.

Background. The version of the TL problem in which the inter-tree edges define a one-to-one mapping between the leaves of the two trees has been extensively studied [4, 5, 8, 2, 10, 7]. We refer to this restricted version as the *single-labeled tanglegram layout (SLTL) problem*. It is known that the SLTL problem is NP-hard [5], even when both the trees are complete [2]. On the positive side, the *one-tree* version of the SLTL problem, where the layout of one of the trees is fixed, can be solved in $O(n \log^2 n)$ time, where n is the number of leaves in the species tree [5]. Moreover, the existence of a planar layout can be verified in linear time [5]. The SLTL is also known to be fixed parameter tractable, where the parameter of interest is the minimum number of crossings in any layout of the two trees [5, 2]. When restricted to complete trees, the SLTL problem is also known to be 2-approximable [2]. Additionally, there is published software that attempts to produce a layout where the number of crossings is small [4, 7].

Our Contribution. In this paper, we define and study the *generalized tanglegram layout (GTL) problem*, in which the number of leaves in the trees may be different and a leaf in either of the trees may share inter-tree edges with multiple leaves in the other tree. The goal again is to produce a layout that minimizes the number of crossings between the inter-tree edges. This generalization of the SLTL problem makes it possible to address not only the gene tree/species tree layout problem, but also those problems in which the inter-tree edges between the trees can be completely arbitrary; such general instances arise in several settings, notably in the analysis of host-parasite co-speciation [11]. The GTL problem has not, to our knowledge, been studied before.

After defining the GTL problem formally, we present two efficient algorithms for its one-tree version, where the layout of one of the trees is fixed. Our algorithms run in time $O(k \log^2 k / \log \log k)$ and $O(kh)$, respectively, where k is the number of inter-tree edges between the two trees, and h is the height of the tree whose layout can change.

Note that for the SLTL problem, k equals n , the number of leaves in each tree. Thus, our first algorithm improves on the best known solution for the one-tree SLTL problem by a factor of $\Theta(\log \log n)$. Based on the result of Fernau et al. [5], we show that the existence of a planar layout (i.e., a layout with no crossings) can be verified in $O(k)$ time. Along the lines of [5,2], we also discuss the fixed parameter tractability of the GTL problem, along with the approximability of a version of the GTL problem which seeks to maximize the number of non-crossing edges.

We have found that, in practice, one-tree algorithms produce layouts that leave considerable room for improvement. Thus, we have designed and implemented fast heuristics that exploit our one-tree algorithm to reduce the number of crossings by rearranging the layouts of both of the input trees. Although our heuristics do not guarantee an optimal layout, we are able to show empirically that they perform quite well for the range of input sizes that one might expect to encounter in practice. To further strengthen this claim, we make use of an integer quadratic programming formulation of the two-tree problem, based on the work of Nöllenburg et al. [10], to solve the GTL problem exactly for realistic input sizes. Our experiments show that our heuristics perform well in practice and, in all but a few cases, our most comprehensive heuristic returns optimum or near-optimum solutions.

2 Basic Notation and Preliminaries

Given a rooted tree T , we write $V(T)$, $E(T)$, and $L(T)$ to denote its node set, edge set, and leaf set, respectively. A node in $V(T)$ that is not a leaf is called an *internal* node. The root node of T is denoted by $rt(T)$. Given a node $v \in V(T)$, $pa(v)$ denotes the parent of v in T , $Ch(v)$ is the set of children of v , and $T(v)$ denotes the subtree of T rooted at v . If two nodes in T have the same parent, they are called *siblings*. T is *fully binary* if every internal node has exactly two children. Throughout this paper, unless otherwise noted, the term *tree* refers to a rooted, fully binary tree.

The generalized tanglegram layout problem. Let S and T be two uniquely leaf-labeled trees such that $L(S) = \{1, \dots, m\}$ and $L(T) = \{1, \dots, n\}$. Furthermore, let $I(S, T)$ be the set of *inter-tree edges* such that $I(S, T) \subseteq L(S) \times L(T)$ and each leaf node of S and T is incident on at least one edge in $I(S, T)$. Given uniquely leaf-labeled trees S and T and the set of inter-tree edges $I(S, T)$, we denote the resulting instance of the GTL problem by $\langle S, T, I(S, T) \rangle$.

Given a tree T , we say that a linear order τ on $L(T)$ is *compatible* with T if, for each $v \in V(T)$, the leaves in $T(v)$ form an interval (i.e., appear in a continuous block) in τ . We write $u <_{\tau} v$ to mean that $u \in L(T)$ appears before $v \in L(T)$ in the order τ .

Given compatible linear orders σ and τ on trees S and T respectively, the *number of crossings* between σ and τ with respect to $I(S, T)$, denoted $cr(\sigma, \tau, I(S, T))$, is defined to be $|\{(u, v) \in I(S, T) : \text{either } u <_{\tau} v, \text{ and } v <_{\sigma} u, \text{ or } u <_{\sigma} v, \text{ and } v <_{\tau} u\}|$.

Problem 1 (GTL Problem). *Given an instance $\langle S, T, I(S, T) \rangle$, find compatible linear orders σ and τ on trees S and T , respectively, such that $cr(\sigma, \tau, I(S, T))$ is minimized.*

Next, we define a restricted version of the GTL problem in which the linear order for one of the input trees is fixed. We call this restricted problem the *One-Tree Generalized*

Tanglegram Layout (OT-GTL) Problem. Given uniquely leaf-labeled trees S and T , the set of inter-tree edges $I(S, T)$, and a compatible linear order τ on T , we denote the resulting instance of the OT-GTL problem by $\langle S, T, I(S, T), \tau \rangle$.

Problem 2 (OT-GTL Problem). *Given an instance $\langle S, T, I(S, T), \tau \rangle$, where τ is a compatible linear order on T , find a compatible linear order σ on the tree S such that $cr(\sigma, \tau, I(S, T))$ is minimized.*

3 Solving the OT-GTL Problem

In this section, we provide two algorithms for the OT-GTL problem on the instance $\langle S, T, I(S, T), \tau \rangle$, one with time complexity $O(k \log^2 k / \log \log k)$ and the other with $O(kh)$, where k denotes the cardinality of the set $I(S, T)$ and h is the height of tree S . We also discuss two important special cases of the OT-GTL problem.

With any given planar layout of a tree, we associate a unique linear order. Therefore, when talking about the tree S , we assume that the tree is drawn in the plane with the root node on top and leaves on the bottom. The unique linear order associated with S is then given by the left-to-right order of the leaf labels. Similarly, when talking about the tree T , we assume that the tree T is drawn in the plane with the root at the bottom and the leaves on the top. The unique linear order associated with T is then given by the left-to-right order of the leaf labels. Also observe that, under this setting, every linear order compatible with S (T) defines a unique planar layout for S (T). Thus, if we rotate the layout shown in Figure 1 clockwise by 90 degrees then the tree on the top would be S and the one on the bottom would be T .

We need some notation. Let σ denote the linear order corresponding to some planar layout of S . For any $v \in V(S(v))$, $I(S(v), T)$ denotes the set $\{(u, v) \in I(S, T) : u \in L(S(v))\}$. We define the number of crossings $cr(\sigma, \tau, I(S, T), v)$ at node v to be $|\{(u, v) \in I(S(v), T) : \text{either } u <_{\tau} v, \text{ and } v <_{\sigma} u, \text{ or } u <_{\sigma} v, \text{ and } v <_{\tau} u\}|$.

Let $\overline{\sigma}_v$ denote the linear order corresponding to the planar layout of S obtained by starting with the planar layout corresponding to σ and then swapping the left and right child at the internal node $v \in V(S)$.

The OT-GTL problem seeks to re-draw the tree S such that the linear order σ_{opt} associated with the new layout minimizes the number of crossings $cr(\sigma_{opt}, \tau, I(S, T))$. The task then is to decide, at each internal node, which one of its two children is to be the left child and which one the right child. It is easy to show that this decision can be taken independently at each internal node, irrespective of the decision at the other nodes. Thus, our algorithm starts with a planar layout of S , with the corresponding linear order σ , and then computes, at each internal node v of S , the values $cr(\sigma, \tau, I(S, T), v)$, and $cr(\overline{\sigma}_v, \tau, I(S, T), v)$. If $cr(\sigma, \tau, I(S, T), v) > cr(\overline{\sigma}_v, \tau, I(S, T), v)$, then we swap the left and right child of v . Once this is done at all internal nodes of S , the compatible linear order associated with the new planar layout of S gives the required solution.

A note on handling the input. Recall that $I(S, T)$ is the set of inter-tree edges. Our algorithms assume that given any leaf node in S (or T), we can access all its p neighbors in the other tree within $O(p)$ time. This can be easily accomplished by associating with each leaf node a set of its neighbors in the other tree. It is possible to construct all

these sets within $O(k)$ time. Since all our algorithms require $\Omega(k)$ time, we ignore this additional additive factor. Note, however, that if the leaf labels are arbitrary (and not, as we assume, $\{1, \dots, n\}$ and $\{1, \dots, m\}$), then handling the input could require $\Theta(k \log(nm))$ time in the worst case.

3.1 An $O(k \log^2 k / \log \log k)$ -Time Algorithm

Our algorithm uses a data structure Ψ for the *subset rank* problem [3]. Such a data structure allows one to maintain a subset $A \subseteq \{1, \dots, n\}$ under the following operations: *Insert*(i, Ψ), which inserts i into Ψ , *Delete*(i, Ψ), which deletes i from Ψ , and *Rank*(i, Ψ), which, given some $i \in A$, returns the number of elements in A that are less than or equal to i . It is known that each of these operations can be performed in $O(\log n / \log \log n)$ time [3].

We will assume, without any loss of generality, that the initial layout of S is such that at each internal node the number of inter-tree edges incident on leaves of the left subtree is at least as large as the number of inter-tree edges incident on leaves of the right subtree. The linear order corresponding to this layout of S is σ . Our algorithm computes, at each internal node v of S , the value $cr(\sigma, \tau, I(S, T), v)$. Later we explain how a slight modification allows our algorithm to compute the value $cr(\overline{\sigma}_v, \tau, I(S, T), v)$ as well. The algorithm proceeds in three different steps:

Step 1: Modify the tree S into a tree S' as follows: For each $x \in L(S)$, let $A(x)$ denote the set of leaf node neighbors of x according to the set of inter-tree edges $I(S, T)$. For each $x \in L(S)$, replace the leaf x with an arbitrary tree, X on the leaf set $A(x)$ such that the linear order associated with X is a subsequence of the order τ . We refer to this modified version of S as S' . Observe that S' need not be uniquely leaf-labeled, and that the number of leaf nodes in S' must be exactly k (i.e., $|I(S, T)|$). Observe, also, that all the internal nodes in S must be internal nodes in S' as well. The inter-tree edges between S' and T now simply connect those leaf nodes of S' and T that have the same labels. Now, we uniquely relabel the tree S' so that the compatible linear order σ' associated with the layout of S' becomes the ordered list $(1, 2, \dots, k)$. The set $I(S', T)$ of inter-tree edges between S' and T is correspondingly updated. Note that each leaf node in S' must be incident on exactly one edge of $I(S', T)$.

This step reduces the instance $\langle S, T, I(S, T), \tau \rangle$ of the OT-GTL problem into the instance $\langle S', T, I(S', T), \tau \rangle$. The following lemma relates these two instances.

Lemma 1. *Given S and S' and any internal node $v \in V(S)$, if v' is the corresponding internal node in S' , then $cr(\sigma, \tau, I(S, T), v) = cr(\sigma', \tau, I(S', T), v')$.*

Thus, to solve the OT-GTL problem on instance $\langle S, T, I(S, T), \tau \rangle$ it is sufficient to solve it on the instance $\langle S', T, I(S', T), \tau \rangle$.

Step 2: We start with the planar layout of S' corresponding to the linear order $\sigma' = (1, 2, \dots, k)$ and modify the tree T into a tree T' as follows: For each $x \in L(T)$, let $B(x)$ denote the set of leaf node neighbors of x according to the set of inter-tree edges $I(S', T)$. For each $x \in L(T)$, replace the leaf x with an arbitrary tree, X on the leaf set $B(x)$ such that the linear order associated with X is a subsequence of the order σ' . We refer to this modified version of T as T' , and the corresponding linear order as τ' . The

set of inter-tree edges, denoted by $I(S', T')$ now simply connects those leaf nodes in S' and T' that have the same labels. Next, we traverse through S' in post order, and store at each internal node of S' , the range of leaf labels in the subtree rooted at that node. Note: This range can be completely specified by two “bounding” integers.

This step reduces the instance $\langle S', T, I(S', T), \tau \rangle$ of the OT-GTL problem into the instance $\langle S', T', I(S', T'), \tau' \rangle$. The following lemma relates these two instances.

Lemma 2. *Given S' , T , and T' , and any internal node $v \in V(S')$, we must have $cr(\sigma', \tau, I(S', T), v) = cr(\sigma', \tau', I(S', T'), v)$.*

It is worth observing that both S' and T' are uniquely leaf labeled, $L(S') = L(T') = \{1, \dots, k\}$, and each leaf node in S' and T' is incident on exactly one edge of $I(S', T')$. Thus, by performing steps 1. and 2. we have effectively reduced an instance of the OT-GTL problem into an instance of the one-tree SLTL problem. Our algorithm on this simplified instance $\langle S', T', I(S', T'), \tau' \rangle$ of the OT-GTL problem proceeds as follows.

Step 3: Consider the path from the root of S' to the left most leaf node (in the layout corresponding to σ'). We now compute the value $cr(\sigma', \tau', I(S', T'), v)$ at each internal node v along this path by calling Procedure COMPUTECROSSINGS on S' , σ' and τ' . Remove all the nodes along this left most path. This decomposes S' into a forest of subtrees each having at most $k/2$ leaves. Also break up the linear order τ' into separate linear orders, each corresponding to a particular subtree and restricted to its leaf set. Now apply step 3 of this algorithm recursively on each of these subtrees.

Procedure COMPUTECROSSINGS: This procedure takes as input a tree S' drawn according to σ' and a linear order τ' , and computes the value $cr(\sigma', \tau', I(S', T'), v)$ at each internal node v along the path P from the root to the left most leaf of S' . Procedure COMPUTECROSSINGS uses the subset rank data structure as follows: Given a leaf x in the tree S' , suppose x appears in the right subtree of an internal node y on P . Let z be the largest leaf label in the left subtree of y . Observe that all the leaves in S' with a label smaller than z must be in the left subtree of y . We will use the subset rank data structure to find the number of nodes that appear after the node x in the linear order τ' , whose labels are smaller than or equal to z . Each such node, when paired with x , must be a crossing pair at node y . It can be shown that procedure COMPUTECROSSINGS has time complexity $O(l \log l / \log \log l)$, where $l = |L(S')|$. Further details are omitted from this extended abstract.

Our three-step algorithm computes at each internal node $v \in V(S')$, the value $cr(\sigma', \tau', I(S', T'), v)$ in $O(k \log^2 k / \log \log k)$ time. In light of Lemmas 1 and 2, this immediately yields the value $cr(\sigma, \tau, I(S, T), v)$ for each internal node $v \in V(S)$.

The following lemma shows how our algorithm can also compute, at each internal node $v \in V(S)$, the value $cr(\overline{\sigma}_v, \tau, I(S, T), v)$, yielding Theorem 1.

Lemma 3. *Let $\overline{\tau}$ denote the linear order obtained by reversing τ . Then, for any $v \in V(S)$, we must have $cr(\overline{\sigma}_v, \tau, I(S, T), v) = cr(\sigma, \overline{\tau}, I(S, T), v)$.*

Theorem 1. *The OT-GTL problem can be solved in $O(k \log^2 k / \log \log k)$ time.*

3.2 An $O(kh)$ -Time Algorithm

We now show that it is possible to solve the OT-GTL problem in $O(kh)$ time, where h is the height of the tree S . This algorithm asymptotically outperforms our $O(k \log^2 k / \log \log k)$ -time algorithm when the tree S is (roughly) balanced.

The main idea is to spend at most $O(k)$ time for all the nodes at any fixed level of S . The algorithm follows:

1. Uniquely relabel the tree S so that the compatible linear order σ associated with the layout of S becomes the ordered list $(1, 2, \dots, m)$. The set $I(S, T)$ of inter-tree edges between S and T is correspondingly updated.
2. Modify the tree T into a tree T' as shown in Step 2. of the $O(k \log^2 k / \log \log k)$ -time algorithm. The set of inter-tree edges, denoted by $I(S, T')$, now simply connects those leaf nodes in S and T' that have the same labels.
3. Traverse through S in post order, and store at each internal node the range of leaf labels in the subtree rooted at that node. Note: This range can be completely specified by two “bounding” integers.
4. Let $x = rt(S)$. In the linear order τ' , for each element i , one can now determine in constant time whether i is in the left or the right subtree of x in the layout (according to σ) of S . This can be used to compute the value $cr(\sigma, \tau', I(S, T'), x)$ as follows:
 Consider each element i of τ' in order. If i is in the left subtree of x then do nothing and skip to the next i . If i is in the right subtree of x , then, by using counters (after an initial $O(k)$ preprocessing step), we can obtain in $O(1)$ time the number of elements j that occur after i in τ' such that j is in the left subtree of x . Set $cr(\sigma, \tau', I(S, T'), x)$ to be the sum of these values over all i in τ' .
5. Split the linear order τ' into two linear orders, one containing all the leaves from the left subtree of x and the other the leaves from the right subtree. Recursively repeat steps 4 and 5 of this algorithm on the subtrees $S(u)$ and $S(v)$, where $u, v \in Ch(x)$.

Our algorithm computes at each internal node $v \in V(S)$, the value $cr(\sigma, \tau', I(S, T'), v)$ in $O(kh)$ time. The following lemma relates these values to the ones we actually want to compute, and yields Theorem 2.

Lemma 4. *Given S, T and T' , and any internal node $v \in V(S)$, we must have $cr(\sigma, \tau, I(S, T), v) = cr(\sigma, \tau', I(S, T'), v)$.*

Theorem 2. *The OT-GTL problem can be solved in $O(kh)$ time, where h is the height of the tree S .*

3.3 Interesting Special Cases

We discuss two special cases of the OT-GTL problem. The first consists of those instances in which the set of inter-tree edges $I(S, T)$ is such that each leaf node of S and T is incident on exactly one edge in $I(S, T)$. This is exactly the SLTL problem discussed in the Introduction. The one-tree SLTL problem has been studied in [54]. Observe that, in this case, we must have $|L(S)| = |L(T)| = |I(S, T)|$. The best known algorithm for this problem runs in $O(n \log^2 n)$ time [5], where $n = |L(S)|$

$= |L(T)|$. Our $O(k \log^2 k / \log \log k)$ time algorithm for the OT-GTL problem becomes an $O(n \log^2 n / \log \log n)$ -time algorithm when restricted to this case.

The second case consists of those instances in which the set of inter-tree edges $I(S, T)$ is such that each leaf node of S is incident on exactly one edge in $I(S, T)$. This restricted version of the OT-GTL problem arises naturally when one or more gene trees must be visually compared with a species tree. As seen in the Introduction, species trees have leaves that are labeled uniquely, while each gene trees may have several leaves with the same leaf label. In this scenario, one wishes to produce a planar layout of each given gene tree aligned with the species tree, such that if one were to draw edges between the corresponding leaf labels of the gene tree the species tree, the number of crossings between these edges is minimized. Thus, the gene tree is the tree S and the species tree is the tree T . Note that, in this case, $|I(S, T)| = |L(S)|$. Our $O(k \log^2 k / \log \log k)$ time algorithm for the OT-GTL problem becomes an $O(m \log^2 m / \log \log m)$ -time algorithm when restricted to this case, where $m = |L(S)|$. Likewise, our $O(kh)$ -time algorithm yields an $O(mh)$ time algorithm.

4 The GTL Problem

The GTL problem is NP-hard, even for simpler special cases [52]. Nöllenburg et al. [10] gave an integer quadratic programming (IQP) based exact solution for the restricted version of the GTL problem in which the set of inter-tree edges defines a one-to-one mapping between the leaves of S and T . Their IQP based approach extends easily to exactly solve the GTL problem as well.

Recall that our $O(k \log^2 k / \log \log k)$ -time algorithm converts the instance $\langle S, T, I(S, T), \tau \rangle$ of the OT-GTL problem into the instance $\langle S', T', I(S', T'), \tau' \rangle$ of the one-tree SLTL problem, before applying Step 3. It can be shown that $cr(\sigma, \tau, I(S, T)) = cr(\sigma', \tau', I(S', T'))$.

Fixed parameter tractability and approximability. Since we have $cr(\sigma, \tau, I(S, T)) = cr(\sigma', \tau', I(S', T'))$, Fernau et al.'s FPT algorithm for the SLTL problem [5], which runs in time $O^*(c^p)$, where c is a constant and p is the number of crossings in an optimal layout of the two trees, yields an $O^*(c^p)$ -time algorithm for the GTL problem as well.² The GTL problem is thus fixed parameter tractable. A similar argument shows that, based on the result of Buchin et al. [2], a dual version of the GTL problem, which seeks to maximize the number of non-crossing edges, admits a 0.878-approximate algorithm.

Existence of a planar layout. Fernau et al. [5] showed that, for the SLTL problem, the existence of a planar layout can be verified with-in $O(n)$ time, where $n = |L(S)| = |L(T)|$. It can be shown that the instance $\langle S, T, I(S, T), \tau \rangle$ of the OT-GTL problem can be converted into the instance $\langle S', T', I(S', T'), \tau' \rangle$ of the SLTL problem with-in $O(k)$ time. Thus, since $cr(\sigma, \tau, I(S, T)) = cr(\sigma', \tau', I(S', T'))$, we have an $O(k)$ -time algorithm to verify the existence of a planar layout for the GTL problem.

Heuristics. We propose three different heuristic approaches for the GTL problem, which utilize our fast exact algorithms for the OT-GTL problem.

² The O^* notation ignores the polynomial component of the fixed parameter algorithm.

Alternating Heuristic (AH): This heuristic first optimizes the layout of tree S by solving the OT-GTL problem, then optimizes the layout of tree T with respect to the new layout of S , then optimizes S again, and so on. Thus, after each iteration the total number of crossings decreases. The heuristic terminates when there is no further reduction in the crossing value.

Local-Search Heuristic (LH): This is a local search based hill-climbing heuristic in which the local neighborhood is defined by the set of all layouts of T obtained from the layout of T corresponding to τ by swapping the left and right subtrees at an internal node. Each of these $O(n)$ trees in the neighborhood is then evaluated by solving the OT-GTL problem on the tree S with respect to that tree. Among these $O(n)$ layouts of T , the one which enables the OT-GTL algorithm (when run on tree S) to find the lowest crossing value, is set to be the new layout of T for the next local search step. The algorithm terminates when no better layouts can be found during the local search step.

Local-Search Alternating Heuristic (LAH): This heuristic consists of running the Local-Search Heuristic as described above, until it terminates, and then reversing the roles of trees S and T and running the Local-Search heuristic again, and so on. The heuristic terminates when no further improvements in the layout can be observed by performing the role reversal.

Let us assume that we use an $O(x)$ time algorithm to solve the OT-GTL problem. Then, the time complexity of (i) AH is $O(ax)$ where a is the number of iterations, (ii) LH is $O(bnx)$ where b is the number of local search steps, and (iii) LAH is $O(cx(m+n))$ where c is the total number of local search steps performed.

5 Experimental Evaluation

To test the effectiveness and utility of our heuristics for the GTL problem, we implemented and tested them on three kinds of datasets: (i) randomly generated input instances, (ii) gene trees/species trees built using a simple probabilistic model for gene-duplication/loss, and (iii) a real world gene tree/species tree dataset. To evaluate the performance of our heuristics, on each dataset, we compute the corresponding performance ratio $\rho = (cr + 1)/(OPT + 1)$, where cr denotes the crossing value for the heuristic and OPT the number of crossings in an optimum layout.

Our heuristics use our fast exact algorithms for the OT-GTL problem. For our experiments, we chose to use the $O(kh)$ -time algorithm for the OT-GTL problem³. This algorithm, as well as the three heuristics were implemented in Python and all experiments we performed on an Intel Core2 Duo 2.4 GHz PC. To test the performance of our heuristics, we also used the IQP based approach to solve the GTL problem optimally. The IQP was solved using the mathematical programming software CPLEX.

Random input instances. We first tested the performance of our heuristics on trees with arbitrary binary topologies and arbitrary inter-tree edges. These instances were

³ Our decision was based on the observation that the $O(kh)$ -time algorithm seems to outperform the $O(k \log^2 k / \log \log k)$ -time algorithm on the input instance sizes used in our experiments.

Table 1. Performance of our heuristics on randomly generated input instances. For each n , the table shows (i) the number k of inter-tree edges, (ii) the number of crossings in a random layout, (iii) the number of crossings, the number of iterations performed, and the time, in seconds, required to produce the layout, for the one-tree case (depicted by 1T) and for the three heuristics AH, LH, and LAH, and (iv) the number of crossings in an optimal layout. All values have been averaged over 10 runs. Note that we have not reported the number of iterations for LAH; these were always observed to be between 2 and 3 times the number of iterations for LH. Due to excessive running times, we did not compute optimal solutions for trees with $n > 50$, and did not apply LH and LAH on trees with $n > 200$.

Input			1T		AH			LH			LAH		OPT
n	k	cr	cr	time	cr	iter	time	cr	iter	time	cr	time	cr
10	11	24.9	13.5	0.00	7.9	3.1	0.00	6.2	3.4	0.01	5.8	0.04	5.8
20	23	113.3	85.8	0.00	61.6	3.2	0.00	57.5	8.5	0.11	57.1	0.22	56.8
30	34	254.9	209.9	0.00	164.9	3.7	0.00	158.0	13.4	0.43	157.8	0.64	157.1
40	46	500.6	406.6	0.00	339.5	3.8	0.01	324.1	16.5	1.07	323.7	1.51	322.9
50	57	824.3	623.6	0.00	527.9	3.4	0.01	513.6	18.4	2.03	512.8	2.84	510.7
100	115	3283.3	2765.1	0.00	2482.7	3.7	0.04	2431.7	42.0	23.58	2431.5	26.99	-
200	230	12868.7	11806.9	0.01	10938.9	3.9	0.10	10670.9	83.8	226.53	10668.7	249.74	-
400	460	52139.5	48623.3	0.03	45904.9	3.9	0.29	-	-	-	-	-	-
800	920	211531.8	199728.2	0.12	193667.0	4.1	1.03	-	-	-	-	-	-

generated as follows: We created two random binary trees, both on n leaves and established a random one-to-one correspondence between the leaf sets of the two trees.⁴ Next, we created an instance of the GTL problem by adding an additional $\lfloor n \cdot 15/100 \rfloor$ randomly selected inter-tree edges. We performed experiments for values of n ranging from 10 to 800. The results of our experiments are depicted in Table 5. The performance ratio ρ , averaged over all $n \in \{10, 20, 30, 40, 50\}$, for one-tree (1T), AH, LH, and LAH, respectively, are 1.61, 1.12, 1.02, and 1.003. It can be seen that LAH performs remarkably well: Out of the 50 input instances for which the optimum solution was known, LAH found an optimum layout in 41 instances. AH also performs quite well and produces an optimized layout almost instantaneously even for large input instances.

Simulated gene trees/species trees. Next, we tested the performance of our heuristics on simulated datasets created by using a simplified birth-death process that mimics gene duplication and loss (see, for example, [11]). To build our trees, we first generated a random binary tree S with n leaves. We then generated a simulated gene tree based on S according to the following probabilistic scheme: At each internal node v in S , the subtree $S(v)$ either duplicates with probability d , is lost with probability r , or remains intact with probability $1 - d - r$. For our experiments we chose d and r to be 0.1 and 0.12 respectively. The value of n ranged from 10 to 800. Table 2 depicts the results of our experiments. The performance ratio ρ , averaged over all $n \in \{10, 20, 30, 40, 50\}$, for one-tree (1T), AH, LH, and LAH, respectively, are 1.33, 1.12, 1.0004, and 1.0004. Both LH and LAH perform equally well in this case; in particular, out of the 50 input instances for which an optimum solution was known, both LH and LAH found an optimum layout in 48 instances.

⁴ Each random tree was created by recursively, and randomly, bi-partitioning its set of leaves.

Table 2. Performance of our heuristics on simulated gene trees/species trees. The layout is identical to that of Table 5 and m denotes the number of leaves in the gene tree.

Input			1T		AH			LH			LAH		OPT
n	m	cr	cr	time	cr	iter	time	cr	iter	time	cr	time	cr
10	10.0	15.2	1.2	0.00	1.2	2.0	0.00	1.2	2.0	0.00	1.2	0.02	1.2
20	16.9	44.6	10.5	0.00	6.9	2.1	0.00	5.6	2.4	0.03	5.6	0.10	5.6
30	34.3	390.9	126.8	0.00	125.8	2.1	0.00	92.7	4.5	0.24	92.7	0.52	92.7
40	44.0	631.1	110.6	0.00	109.7	2.2	0.00	83.5	4.1	0.34	83.5	0.79	83.1
50	70.0	2098.1	913.9	0.00	845.2	2.5	0.01	767.5	7.1	1.98	767.5	3.46	767.4
100	93.5	2757.4	815.9	0.00	748.0	2.5	0.02	515.0	8.1	5.04	515.0	7.89	-
200	269.9	31671.8	9128.1	0.02	8407.6	2.8	0.13	7012.0	22.6	172.51	7012.0	219.27	-
400	360.5	35941.7	2942.7	0.02	2823.5	3.1	0.17	-	-	-	-	-	-
800	741.6	153141.1	29695.2	0.09	27854.6	3.0	0.59	-	-	-	-	-	-

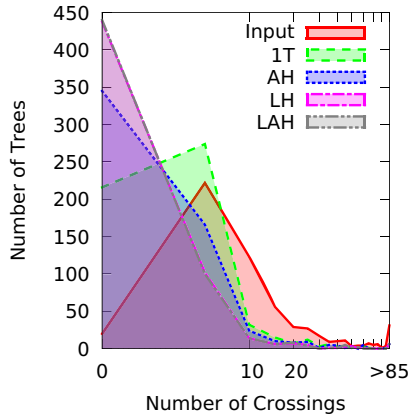


Fig. 2. Distribution of crossing values for the empirical gene tree/species tree dataset after applying various tanglegram layout algorithms. The distributions for LH and LAH are almost identical.

Empirical Dataset. Finally, we tested our heuristics on a real-world empirical dataset [12] on Angiosperms. This data set consists of 588 gene trees with number of leaves ranging from 4 to 94 and a species tree with 7 taxa. On the unoptimized input, the average performance ratio ρ for each gene tree/species tree pair is 10.12, while the corresponding values for 1T, AH, LH, and LAH are 2.32, 1.62, 1.02 and 1.0005 respectively. In particular, there were only one and six input instances, respectively, for which LAH and LH did not find an optimum layout. On all the 588 input instances together, AH took 0.21 seconds, LH took 2.4 seconds, and LAH took a total of 12.5 seconds. Figure 5 depicts the distribution of crossing values after applying the one-tree algorithm and the three heuristics. Indeed, after applying LH and LAH to the dataset, a majority of the input instances show no crossings, a dramatic improvement over the initial layout.

Comparison against other heuristics. Nöllenburg et al., in [10], introduced a heuristic for the SLTL problem. This heuristic has a time complexity of $O(8^h + n^2h)$, where h is the minimum height of the two trees. Since this is exponential in the height of the

tree, they also proposed a branch-and-bound algorithm, referred to as *rec-split-bb*, to curb the time complexity of the heuristic. As shown earlier, the GTL problem can be reduced to an instance of the SLTL problem, and consequently, it is possible to use *rec-split-bb* as a heuristic for the GTL problem. Though *rec-split-bb* seems to be quite efficient in practice, there is no polynomial bound on its running time. All our heuristics are guaranteed to terminate within polynomial time. Our heuristics LH and LAH also seem to perform at least as well as the *rec-split-bb* heuristic (based on the experimental results reported in [10] for random general input trees), if not significantly better.

Acknowledgements. The authors wish to thank Dan Gusfield for introducing them to the tanglegram layout problem and for many helpful discussions, and Balaji Venkatchalam for bringing relevant previous work to their attention. This work was supported in part by NSF grant no. 0334832.

References

1. Arvestad, L., Berglund, A.-C., Lagergren, J., Sennblad, B.: Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. In: RECOMB, pp. 326–335 (2004)
2. Buchin, K., Buchin, M., Byrka, J., Nöllenburg, M., Okamoto, Y., Silveira, R.I., Wolff, A.: Drawing (Complete) Binary Tanglegrams: Hardness, Approximation, Fixed-Parameter Tractability. In: Graph Drawing (2008)
3. Dietz, P.F.: Optimal algorithms for list indexing and subset rank. In: Dehne, F., Santoro, N., Sack, J.-R. (eds.) WADS 1989. LNCS, vol. 382, pp. 39–46. Springer, Heidelberg (1989)
4. Dwyer, T., Schreiber, F.: Optimal leaf ordering for two and a half dimensional phylogenetic tree visualisation. In: InVis. au, pp. 109–115 (2004)
5. Fernau, H., Kaufmann, M., Poths, M.: Comparing trees via crossing minimization. In: Ramanujam, R., Sen, S. (eds.) FSTTCS 2005. LNCS, vol. 3821, pp. 457–469. Springer, Heidelberg (2005)
6. 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)
7. Holten, D., van Wijk, J.J.: Visual comparison of hierarchically organized data. In: 10th Eurographics/IEEE-VGTC Symposium on Visualization (EuroVis 2008) (2008)
8. Lozano, A., Pinter, R.Y., Rokhlenko, O., Valiente, G., Ziv-Ukelson, M.: Seeded tree alignment and planar tanglegram layout. In: Giancarlo, R., Hannenhalli, S. (eds.) WABI 2007. LNCS (LNBI), vol. 4645, pp. 98–110. Springer, Heidelberg (2007)
9. Maddison, W.P.: Gene trees in species trees. *Systematic Biology* 46(3), 523–536 (1997)
10. Nöllenburg, M., Holten, D., Völker, M., Wolff, A.: Drawing binary tanglegrams: An experimental evaluation. In: CoRR, abs/0806.0928 (2008); an updated version of this manuscript will appear in: The proceedings of ALENEX 2009
11. Page, R.D.M. (ed.): *Tangled Trees: Phylogeny, Cospeciation, and Coevolution*. University of Chicago Press (2002)
12. Sanderson, M., McMahon, M.: Inferring angiosperm phylogeny from EST data with widespread gene duplication. *BMC Evolutionary Biology* 7(suppl. 1), S3 (2007)
13. Syvanen, M.: Cross-species gene transfer; implications for a new theory of evolution. *J. Theor Biol.* 112 (1985)

Three-Dimensional Multimodality Modelling by Integration of High-Resolution Interindividual Atlases and Functional MALDI-IMS Data

Felix Bollenbeck¹, Stephanie Kaspar², Hans-Peter Mock², Diana Weier²,
and Udo Seiffert¹

¹ Fraunhofer Institute for Factory Operation and Automation IFF, D-39106
Magdeburg, Germany

Tel.: +49 (0)391 4090 790; Fax: +49(0)391 409093 790,

felix.bollenbeck@iff.fraunhofer.de

www.iff.fraunhofer.de

² Leibniz Institute of Plant Genetics and Crop Plant Research, D-06466 Gatersleben,
Germany

Abstract. We present an approach for the analysis of phenotypic diversity in morphology and internal composition of biological specimen by means of high resolution 3-D models of developing barley grains. Three-dimensional histological structures are resolved by reconstructing specimen from large stacks of serially sectioned material, which is a preliminary for the spatial assignment of key tissues in differentiation. By sampling and constructing models at different developmental time steps from multiple individuals, we address two aims in a computational phenomics context: i) Generation of averaging atlases as structural references for integration of functional data, and ii) building the basis for a mathematical model of grain morphogenesis. We have established an algorithmic pipeline for automated processing of large image stacks towards phenotypic 3-D models and data-integration, comprising registration, multi-label segmentation, and alignment of functional measurements. The described algorithms allow high-throughput reconstruction and tissue recognition of datasets comprising thousands of images. The usefulness of the approach is demonstrated by automated model generation, allowing volumetric measurements of tissue composition, three-dimensional analysis of diversity, and the integration of MALDI-IMS data by mutual information based registration, which is a significant contribution to a systematic analysis of differentiation and development.

Keywords: Plant Phenotyping, 3-D Modelling, Computational Phenomics, Multimodality Registration.

1 Introduction

Plant species provide an immense diversity in morphology, growth habits, and useful traits ranging from bulk biomass fuels to pharmaceutically important compounds, where crop plants, particularly cereals, are certainly most relevant. The description of plant phenotypes has a great tradition in breeding, in connection

with huge success drastically increasing the world's food production in the last century. Today's high-throughput methods, assaying living systems on all scales and domains will lead to unprecedented enhancement of our insights into plant growth and development.

In the post-genomics era, with huge amounts of experimental data available for specific domains, the aim is to gain a holistic understanding of the complex systems of life by analysis and integration of observations of different assaying techniques and modalities.

Apart from combined investigation of genome, proteome, and metabolome data based on digested material, the observation of intact structure and function or even their dynamics *in vivo* have become widely available with technical advance in the past years. Either scanning, tomographic, or destructive methods have in common to allow the observation of phenomena in their natural three-dimensional extension, which literally gives a new dimension to analysis. The processing of such data often comprises non-trivial signal and image processing steps in reconstruction towards valid and revealing visualizations. While data-preprocessing and -reconstruction are often legitimately delegated to engineers by biologists, the systematic or algorithmic analysis, rather than subjective manual inspection, of reconstructed data itself is of great importance for objectiveness and reproducibility, thereby requiring knowledge of biological background as well as computational and mathematical methods.

In the context of analysis and modelling of systems in 3-D, the representation of inevitable biodiversity amongst observed species is an important topic: For the integration of functional data, most likely sampled from different individuals, averaging or *interindividual* models, reflecting phenotypic diversity are highly desirable. This also holds for the generation of quasi-continuous spatio-temporal developmental models based on interpolation of intermediate stages between physical measurements. Here the identification of common themes and structures, and discrimination from phenotypic diversity is essential to any modelling.

We work in the context of modeling growth and development of barley (*Hordeum vulgare*) grains (fig. 1) in 3-D, generating high resolution atlases of relevant tissues and structures for the integration of functional data based on a multitude of samples covering the observed phenotypic variances of individual specimen.

2 Related Work

While in medical applications, standards for imaging devices and acquisition exist to a large extent, 3-D imaging in plant biology is very much application specific. Non-destructive bio-imaging methods such as *laser scanning microscopy*, *X-ray* or *magnetic resonance imaging* have been employed to generate models on microscopic scales, either by direct intensity-based visualizations or further processing such as tissue or organ labelling, as described in recent publications [10], [15], and [4]. Tomographic and scanning methods have several drawbacks,

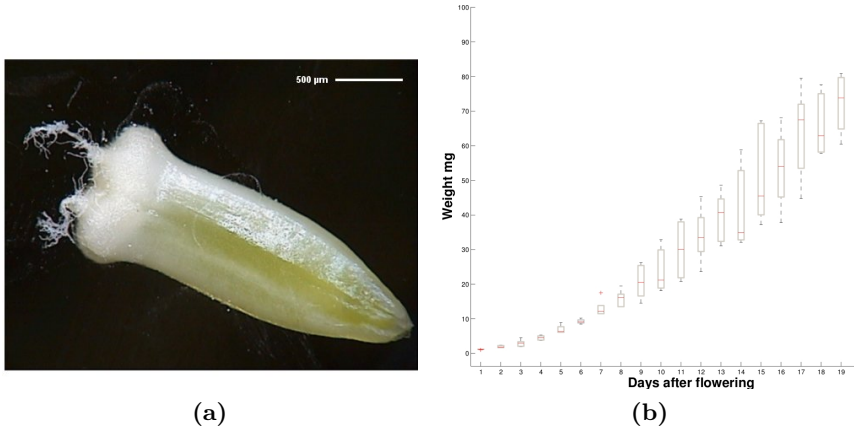


Fig. 1. Barley (*Hordeum vulgare*): **1a**: A developing barley grain 7 days after flowering (DAF). **1b**: Distribution of weights during early grain development, showing large variances in grain phenotypes even under standardized conditions.

particularly in plant systems, delivering coarse spatial resolution for tomographic methods or limited dye penetration caused by cell walls.

Destructive methods overcome limitations in resolution and sampling depth by serially sectioning the object, allowing an analysis of histological structure and functional data, such as gene-specific expression levels as described in [11], but have comprehensive demands in image reconstruction. In vertebrate systems the authors of [21] analyze the development of mice embryos, comprising reconstruction algorithms ([12]) and integration of structural information and gene-expression data in a bioinformatics context ([2]). Algorithms and methods for reconstruction of section data are discussed in detail in [22] while an application in 3-D model generation of plant organs based on individual specimen is described in [9].

Imaging techniques gain more importance in proteome and metabolome analysis also (see [13] for a compact review). MALDI imaging mass spectrometry (MALDI-IMS), as a relatively new technique has received great attention recently (see [8] for an overview), allowing detailed analysis of proteins and small molecules by resolving full MS spectra spatially, which results in new challenges in analysis and visualization.

3 Methods

In this section we describe the automated generation of high-resolution 3-D models from serial section data. We address reconstruction (registration), tissue labelling (segmentation) and integration of multiple samples, for efficient processing of data-sets comprising several thousands of light-microscopic (LM) images, and describe a statistical averaging model as well as the integration of functional data modalities into the obtained 3-D models.

3.1 Data Reconstruction from Section Images

The object of interest is physically destroyed for digitization, whereby the three-dimensional coherence of structures is lost by manual slide mounting and digitization. Since manual or interactive processing is virtually impossible for stacks comprising several thousand images, all images are registered to recompose the intact grain structure, i.e. each image is transformed towards an optimal superposition of structures of all images in the stack.

Automated landmark detection, e.g. using algorithms like SIFT [16] is difficult in biomedical images, especially in the absence of unique structures in histological images. While finding an optimal transform based on direct intensity values is computationally more costly than aligning two sets of coordinates, it can be generally considered less error prone than matching images based on such landmarks which could possibly be mis-identified beforehand.

Intensity Based Registration. In the context of intensity-based image registration a broad range of image-to-image metrics are described in the literature [20]. While statistical and information theoretic metrics [17] address images coming from different modalities (*inter-modality* registration), images $R, T : \Omega \subset \mathbb{R}^2 \mapsto \mathbb{R}$ with similar intensity distributions allow an intuitive and computational cheap metric by direct comparison of quadratic difference intensity magnitude at grid points

$$D(T, R) := \frac{1}{2} \int_{\Omega} (R(x) - T(x))^2 dx. \tag{1}$$

An optimal superposition of all stack images $\mathbf{R} := (R_1, \dots, R_M)$ is found by minimizing the sum of squared differences of images intensities over the space of linear transformations $\varphi = (\varphi_1, \dots, \varphi_M)$ for each image:

$$(R \circ \varphi) := \sum_{i=1}^M \int_{\Omega} (R(x)_{i-1} \circ \varphi_{i-1} - R(x)_i \circ \varphi_i)^2 dx \stackrel{!}{=} \min. \tag{2}$$

Extended Image Metric. Stack registration based on pairwise alignment is critical, since small misalignments are propagated through the whole image stack. By extending the image metric to incorporate intensity values of images within a neighborhood N

$$D(R \circ \varphi) := \sum_{i=1}^M \sum_{j=i-N}^{i-1} \int_{\Omega} (R_j(x) \circ \varphi_j - R_i(x) \circ \varphi_i)^2 dx \stackrel{!}{=} \min \tag{3}$$

a more robust registration is obtained.

Multiresolution Framework. Due to the textured nature of images, the surface of metric values in parameter space is unsmooth, possibly causing the search getting stuck in local optima. Performing registration in a multi-resolution image hierarchy has the advantage of successively improving the registration result by choosing appropriate parameters for each scale, thereby

- Avoiding high metric frequencies using large scales initially
- Optimizing in narrowed search space for smaller scales

while being computationally much more efficient.

Grid Search. Exhaustive search, sampling the whole search space on a regular grid can be considered unfeasible with large images due to computing times, but on the other hand guarantees a global optimum at sufficient grid density. While numerical optimization schemes tend to get stuck in local optima near the initialization point, i.e. flipped images, an exhaustive search on larger scales and smaller resolutions finds the *correct* transform even if metric values differ only slightly.

Numerical Optimization in Transform Space. The global optimum found via an extensive search on a low resolution and large scales is used to initialize the registration on finer level. For further refinement of the registration result, now gradient descent optimization is employed. Depending on the resolution on scale level of the hierarchy, optimization parameters such as step length are annealed when switching to smaller scales.

Gradient descent optimizers are generally susceptible to different scales of the input variables. Rotation and translation have dynamic ranges differing in two magnitudes. Algorithms specifically addressing this problem have been proposed using evolutionary optimization strategies [24], generally showing slower convergence compared with gradient descents. We therefore employ a rescaling of the gradient vector for a gradient descent search.

The employed registration scheme is robust to image distortions, since the annealed step widths on finer levels prohibit to skew the overall alignment.

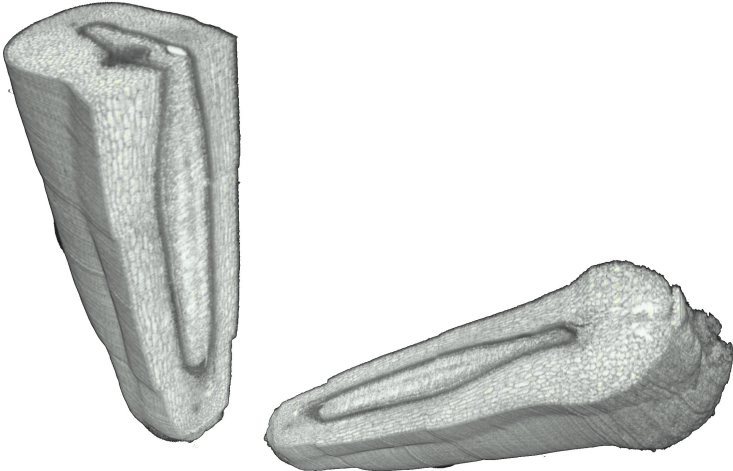


Fig. 2. Grain reconstruction: The intact grain histology is restored by registering the stack of section images using an SSD based image metric within a local range of consecutive z-slices, as a basis for subsequent tissue labelling. The figure shows two perspectives of registered voxel dataset comprising approximately 2,000 slices (4 GB).

3.2 Automated Tissue Recognition

In order to obtain the basis for a quantitative description of tissues and internal structures (see fig. 2), these must be recognized and labelled within section data. While the recognition of multiple tissues in histological data is non-trivial and requiring expert knowledge, automation is highly feasible for objectiveness and time-saving. This classification is a crucial step in the modelling pipeline: Here the raw intensity data is abstracted towards the rationale of the modelling process itself, where labelled voxel-data is the basis for quantification and surface-based modelling of internal structures.

Relevant structures within a section image $I : \Omega \subset \mathbb{R}^2 \mapsto \mathbb{R}^+$ are assigned a unique label $S : I \subset \Omega \mapsto \{1, \dots, M\}$ for M tissues or classes. An automatic segmentation of sections is characterized by several requirements:

- Multiple tissues must be recognized
- Images lack unique structures, edges etc.
- The identification of tissue types needs expert knowledge

Segmentation based on expert created references, or atlases, delivers good segmentation accuracies, when image features are ambiguous. In 3 we describe the segmentation of large stacks of histological section data into multiple classes by intensity driven registration and deformation of reference segmentations, performing equally with image-feature based supervised classifiers like *support vector machines* and *multi layer perceptrons* described in e.g. 9 and 5 while being less computationally costly.

By classification of histological data into three-dimensional tissue mappings of individual morphology are obtained, which can be visualized by iso-surface renderings as in 11 and 9. This is the prerequisite for inter-individually valid models, by integration and fusion of such morphological maps from multiple individuals.

3.3 Statistical 3-D Models of Barley Grains

In the context of histological 3-D models, works so far have neglected biodiversity amongst specimen. A description of diversity amongst multiple objects, allowing the identification of common themes and structures and statistical description of variances in phenotypes during development, provides a meaningful framework for the integration of data acquired from other individuals.

The quantification of inter-individual variability requires the mapping of data into a common reference frame towards the estimation of ubiquitous tissues and regions of varying tissue composition. For spatially resolving internal structures into an averaging model, spatial coordinates are assigned a probability representing a specific tissue or material. Probabilities are estimated from the set of multiple individual segmented histology volumes, which generally vary in orientation and spatial extension. To obtain a transformation invariant to the actual tissue mapping, datasets are registered into a common coordinate frame by standardizing first order moments of the mass-centered intensities of the respective grayscale volumes of sectioned images as suggested in 11 and

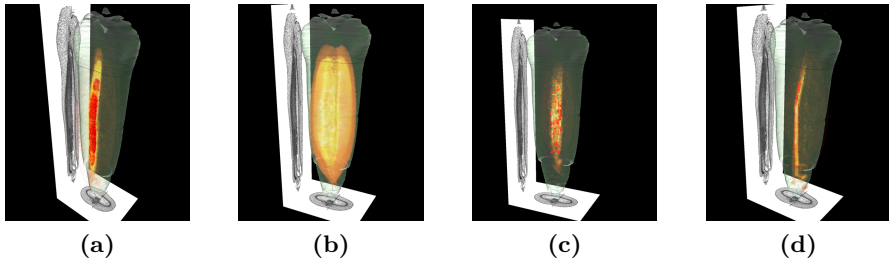


Fig. 3. Visualization of interindividual tissue compositions: Rendering of statistical models displaying ubiquitous regions common to all samples (red surface) and tissue-specific probabilities respectively. [3a](#): Nucellar Projection, [3b](#): Endosperm, [3c](#): Transfer cells, [3d](#): Vascular bundle.

[22](#). Instead of using registration approaches directly maximizing the correspondence of individual label- or grayscale volumes with affine mappings, spline, or free-form deformations, a registration based only on individual image statistics can be considered *un-biased* in terms of leaving the inter-individual variances unaffected.

For each gridpoint $\mathbf{x} \in \Omega \subset \mathbb{R}^3$ and tissue M we estimate a probability for \mathbf{x} belonging to tissue M empirically by

$$p_{\mathbf{x},M} := \frac{1}{|S|} \sum_{i=1}^{|S|} \delta(S_i(\mathbf{x}), M) \quad (4)$$

from segmented datasets S_i .

Thereby, a closed probabilistic description representing the spatial distribution of tissues and materials amongst specimen terms of a mapping $P : \Omega \mapsto \mathbb{R}^M$ is obtained.

While the spatial distribution of tissue probabilities is not based on assumptions of an underlying distribution as with statistical deformation models, it can directly be related to underlying histological information (intensity volumes) as indicated in fig. [3](#).

3.4 Intermodality Registration

Functional data, as different imaging modalities, such as NMRi or MALDI-IMS, generally do not exhibit structures allowing the spatial assignment of tissue types, while such an assignment can generally be considered highly desirable especially when elucidating the mechanism of development and differentiation.

By using multimodality-registration approaches, different assays are registered based on statistical measures: Since two modalities now have different intensity (sec. [3.1](#)) distributions, an image-to-image metric must relate to broader correlation using mutual information based metrics introduced in [7](#) and [25](#). An

optimal registration is obtained by a transformation minimizing the entropy of the joint histograms of pixel intensities:

$$MI(R, T) = \sum_{a,b} p_{RT}(a, b) \log \left(\frac{p_{RT}(a, b)}{p_R(a)p_T(b)} \right). \quad (5)$$

By discretizing the dynamic range to a smaller number of equally sized bins, the estimation of joint probabilities from (joint) image histograms can be reduced to a fixed set of spatial samples being considerably smaller than the full image grid. High-frequency oscillations in the energy surface are avoided by a weighted bin contribution on interpolation grid points of the image domain, using a third order interpolation kernel, as suggested in [18].

For the registration of histological image data with other modalities such as MALDI-IMS, linear transformations are optimized for maximal mutual information using a gradient descent optimizer, as described by [14] with an implementation available in the ITK framework.

4 Results

For this work five developing barley (*hordeum vulgare*) grains at identical developmental stage (7 DAF) with individual weights ranging from 11 to 19mg (based on the observed weight variances, fig. 1b) were embedded, contrasted, sectioned with a microtome and digitized yielding 2,128 to 2,736 slice images, each of size 1200×1600 pixels (spatial res. $1.83 \times 1.83 \mu\text{m}$ 12 bit single channel (approx. 30 GB image data).

Intact individual grains were reconstructed from section images using the proposed registration methods. Fig. 2 shows a volume rendering of a registered individual grain. Reconstructed data was segmented into relevant tissues using the described registration based algorithm with a subset of expert-created reference segmentations.

4.1 3-D Averaging Models

For inter-individual description we registered and joined the segmented data in a common reference, yielding a volume of probabilities for each tissue. Using the probabilistic modelling, we addressed the biological questions (1.) how are specific tissues and relevant material varying and (2.) what are ubiquitous themes amongst individuals. Thus, for an insightful 3-D visualization of probabilistic models or atlases, we are using a combination of two methods:

1. Volume rendering for the spatial distribution of tissue probability values
2. Surface rendering for ubiquitous regions, i.e. $p_{x,M} = 1$

Fig. 3 shows a combined rendering for four maternal and filial tissues known to play an important role in early grain development: *Nucellar Projection* (fig. 3a), *Endosperm* (fig. 3b), *Transfer cells* (fig. 3c), and the *vascular bundle* (fig.

3d). The transparent outer hull of an individual grain and a virtual lateral slice are displayed for better intuition.

Although a strong variance in grain phenotypes within a cultivar even under standardized conditions is observed, as shown by the grain weight variances in fig. **1b**, there are ubiquitous regions for these tissues, where the specific tissues have different variances in their spatial distribution.

As visualized by the volume-rendering of tissue probabilities, this voxel-based description has the advantage of allowing the extraction of surface models at arbitrary probability values, which can be used to predict tissue mappings for new instances.

4.2 Integration of MALDI-IMS Assays

As a relatively new technique, MALDI-IMS allows spatially resolved mass spectroscopy measurements by rasterized laser ionization and detection of matrix coated sample sections (**6**, **19**). By sequentially probing raster points, a 2-D mass spectroscopic image is obtained, where each pixel holds a full mass spectrum, with m/z peak intensities relating to different proteins and metabolites. The spatial distribution of such functional data is particularly insightful, when related to structural properties, such as different tissues or material by means of inter-modality registration, as described in **23** for NMR imaging.

We propose the integration of 2-D MALDI-IMS data into the interindividual 3-D tissue models described above: The integration of spatial models based on histological images and functional measurements allows the analysis of colocalization and gradients over tissues, and thereby the identification of candidate molecules.

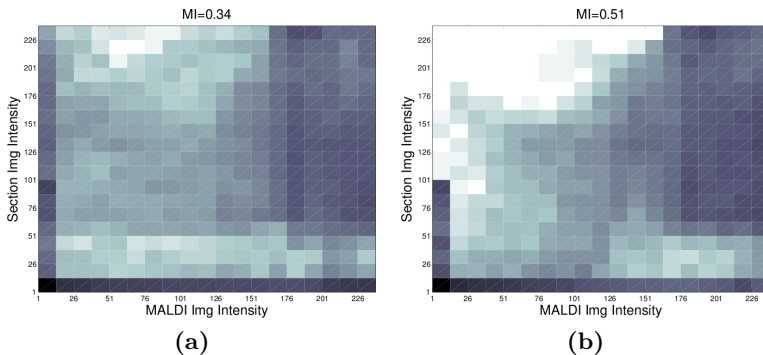


Fig. 4. Integration of structural and functional data: Inter-modality registration of a virtual lateral histological section and a MALDI-IMS scan ($m/z = 9595$). An optimal transformation is found by maximizing MI. Fig. **4a** shows the joint histogram using 20 bins *before* and fig **4b** *after* the transformation (1.82 degrees rotation, translation $x = 21.42\mu m$, translation $y = -49.8\mu m$), resulting in significantly less uniformity of the histogram entries and correspondingly higher MI value.

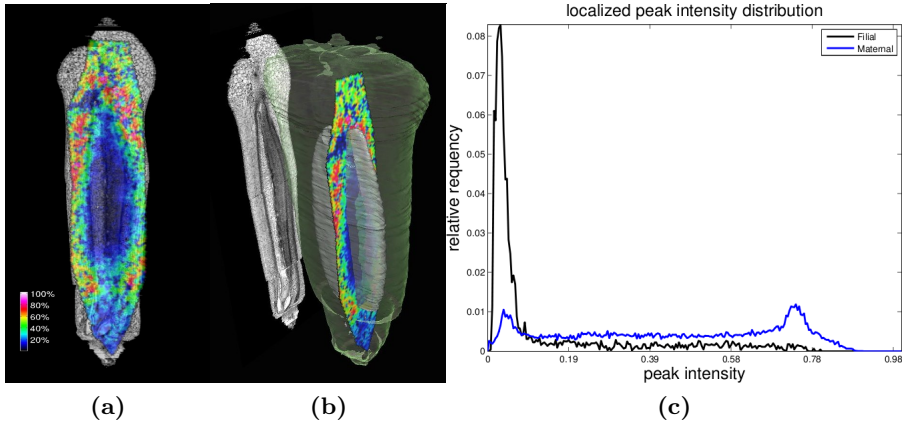


Fig. 5. Integration workflow of MALDI-IMS data into structural models: Fig. 5a shows an image of an MALDI-IMS channel ($m/z=9,595$) and the virtual histological section obtained by MI multimodality registration. Fig. 5b shows the according 3-D rendering of MALDI-IMS peak intensity distribution at the calculated position into the tissue atlas based on histological data. Peak intensities are color encoded from *blue* (low) to *red* (high). A differential localization in maternal (*green*) and filial (*gray*) tissue can be observed. Fig. 5c shows the tissue specific normalized peak intensity distribution for maternal and filial regions, supporting the visual impression that peak intensities are differentially distributed in maternal and filial tissues (rank-sum $p \approx 0$ at 5% significance).

Inter-Modality Registration. MALDI-IMS measurements were taken from a barley grain specimen (7 DAF) from a frozen lateral section on a $30\mu\text{m}$ ionization raster. The full mass spectra were detected for each raster point.

The peak intensity distribution for an unknown molecular ion $[M + H]^+$ at $m/z=9,595$ was sampled from mass spectra, composing a single channel 2-D image. The image was then registered into three-dimensional data maximizing mutual information between the MALDI-IMS image and histological data by gradient descent optimization with linear transformations (rotation and translation), using virtual lateral sections, sampled on an $20\mu\text{m}$ grid from histological data (fig. 2). We used 20 equally sized bins for estimation of joint and marginal probabilities and 10,000 image grid points which were kept fixed during optimization. The highest mutual information value over all image pairs was chosen as the correct positioning. Fig. 4 displays the joint histogram frequency counts $\{q_{ij}\}$ before and after transformation.

Fig. 5b displays a rendering of the aligned MALDI-IMS image at its correct position in a virtual grain, and according histological section as a projection view. The rendered tissue surfaces comprise *maternal* and *filial* types: *Pericarp* (green) and *Endosperm* (gray). Peak intensities are color encoded (*blue* correspond to low *red* to high peak intensities). The visualization promotes the assumption, that the distribution of peak intensities at $m/z=9,595$ is differentially co-localized with these two tissue types.

Localization of MALDI-IMS Peak Intensities. To further investigate the assumption of differential localization, we evaluated the peak intensity distribution within the prevailing maternal and filial tissues: Based on tissue labelling obtained from the registration into the structural 3-D model, peak intensity grid points were mapped to either tissue.

Fig. 5c displays the distribution of tissue specific normalized relative peak intensities. The clustering of intensities near zero for filial tissue indicates that the measured ion is clearly differentially distributed between both tissues (rank-sum p-value of zero at 5% significance), which makes this ion interesting for further biochemical characterization, being a putative actor in development of grains.

5 Discussion and Outlook

In this contribution we have described the automatic generation of interindividual 3-D models of microscopic plant organs. Delivering superior spatial resolution and particularly histological information, models are based on serial section images. We algorithmically address reconstruction and tissue recognition of large image stacks, which is the basis for fast and reproducible processing of multiple specimen for statistically valid models.

Analysis of spatial tissue composition amongst different individuals during growth and development delivers a high value-added for functional assays of tissues exhibiting differential genetic and metabolomic patterns. The usefulness of 3-D models for the integration and analysis of functional measurements is demonstrated by analyzing differential peak patterns in MALDI-IMS data. Here, 3-D models and automated data integration by means of multimodality registration reveal patterns within the observed data, which would have remained obscure in individual analysis. The analysis of spatial as well as temporal gradients of molecules (metabolites, expressed genes, or proteins) is a key feature in understanding development. In this context, works in spatial or spatio-temporal analysis is of high relevance prospectively. The identification of candidate molecules using spatial models and functional data such as full spectrum MALDI-IMS data seems promising, particularly with measurements on a timeline. For such digital morphogenesis, averaging models at different physical measurement are a preliminary.

Acknowledgement

This work is supported by BMBF grant 0313821A. The authors would like to thank Uta Siebert and Birgit Zeike (IPK Gatersleben) for material preparation, Michael Becker (Bruker Daltonik GmbH, Bremen, Germany) for assistance with MALDI-IMS measurements, and Andrea Matros and Winfriede Weschke (IPK Gatersleben) for fruitful discussions and useful comments.

References

1. Alpert, N.M., Bradshaw, J.F., Kennedy, D., Correia, J.A.: The principal axes transformation - a method for image registration. *The Journal of Nuclear Medicine* 31, 1717–1722 (1990)
2. Bard, J.B.L., Baldock, R.A., Davidson, D.R.: Elucidating the genetic networks of development: A bioinformatics approach. *Genome Research* 8(9), 859–863 (1998)
3. Bollenbeck, F., Seiffert, U.: Fast registration-based automatic segmentation of serial section images for high-resolution 3-D plant seed modeling. In: 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2008. ISBI 2008, pp. 352–355 (2008)
4. Bougourd, S., Marrison, J., Haseloff, J.: An aniline blue staining procedure for confocal microscopy and 3-D imaging of normal and perturbed cellular phenotypes in mature *Arabidopsis* embryos. *The Plant Journal* 24, 443–550 (2000)
5. Brüß, C., Strickert, M., Seiffert, U.: Towards automatic segmentation of serial high-resolution images. In: Proceedings Workshop Bildverarbeitung für die Medizin, pp. 126–130 (2006)
6. Chaurand, P., Norris, J.L., Cornett, D.S., Mobley, J.A., Caprioli, R.M.: New developments in profiling and imaging of proteins from tissue sections by MALDI mass spectrometry. *J. Proteome Res.* 5, 2889–2900 (2006)
7. Collignon, A., Maes, F., Delaere, D., Vandermeulen, D., Suetens, P., Marchal, G.: Automated multi-modality image registration based on information theory. In: *Information Processing in Medical Imaging*, pp. 263–274 (1995)
8. Cornett, D.S., Reyzer, M.L., Chaurand, P., Caprioli, R.M.: MALDI imaging mass spectrometry: molecular snapshots of biochemical systems. *Nature Methods* 4, 828–833 (2007)
9. Dercksen, V.J., Brüß, C., Stalling, D., Gubatz, S., Seiffert, U., Hege, H.-C.: Towards automatic generation of 3D models of biological objects based on serial sections. In: Linsen, L., Hagen, H., Hamann, B. (eds.) *Visualization in Medicine and Life Sciences*, pp. 3–25. Springer, Heidelberg (2008)
10. Glidewell, S.M.: NMR imaging of developing barley grains. *Journal of Cereal Science* 43, 70–78 (2006)
11. Gubatz, S., Dercksen, V.J., Brüß, C., Weschke, W., Wobus, U.: Analysis of barley (*Hordeum vulgare*) grain development using three-dimensional digital models. *The Plant Journal* 52(4), 779–790 (2007)
12. Guest, E., Baldock, R.: Automatic reconstruction of serial sections using the finite element method. *Bioimaging* 3(4), 154–167 (1995)
13. Heeren, R.M.A.: Proteome imaging: a closer look at life's organization. *Proteomics* 5(17), 4316–4326 (2005)
14. Ibanez, L., Schroeder, W., Ng, L., Cates, J.: *The ITK Software Guide*, 2nd edn. Kitware, Inc. (2005) ISBN 1-930934-15-7, <http://www.itk.org/ItkSoftwareGuide.pdf>
15. Lontoc-Roy, M., Dutilleul, P., Prasher, S.O., Han, L., Smith, D.L.: Computed tomography scanning for three-dimensional imaging and complexity analysis of developing root systems. *Can. Jour. of Botany* 83(11), 1434 (2005)
16. Lowe, D.G.: Object recognition from local scale-invariant features. In: *International Conference on Computer Vision, Kerkyra, Greece*, vol. 2, pp. 1150–1157 (1999)
17. Maes, F., Collignon, A., Vandermeulen, D., Marchal, G., Suetens, P.: Multimodality image registration by maximization of mutual information. *Trans. on Med. Imaging* 16(2), 187–198 (1997)

18. Mattes, D., Haynor, D.R., Vesselle, H., Lewellyn, T.K., Eubank, W.: Nonrigid multimodality image registration, vol. 4322, pp. 1609–1620 (July 2001)
19. McDonnell, L.A., Piersma, S.R., Altelaar, A.F.M., Mize, T.H., Luxembourg, S.L., Verhaert, P.D.E.M., van Minnen, J., Heeren, R.M.A.: Subcellular imaging mass spectrometry of brain tissue. *Journal of Mass Spectrometry* 40(2), 160–168 (2005)
20. Modersitzki, J.: *Numerical Methods for Image Registration*. Oxford University Press, Oxford (2004)
21. Ringwald, M., Baldock, R.A., Bard, J., Kaufman, M.H., Eppig, J.T., Richardson, J.E., Nadeau, J.H., Davidson, D.: A database for mouse development. *Science* 265, 2033–2034 (1994)
22. Schmitt, O., Modersitzki, J., Heldmann, S., Wirtz, S., Fischer, B.: Image registration of sectioned brains. *Int. J. Comput. Vision* 73(1), 5–39 (2006)
23. Sinha, T.K., Khatib-Shahidi, S., Yankeelov, T.E., Mapara, K., Ehtesham, M., Cornett, D.S., Dawant, B.M., Caprioli, R.M., Gore, J.C.: Integrating spatially resolved three-dimensional MALDI IMS with in vivo magnetic resonance imaging. *Nature Methods* 5, 57–59 (2008)
24. Styner, M., Gerig, G.: Evaluation of 2D/3D bias correction with 1+1 ES-optimization – Technical Report 179. Technical report, Image Science Lab, ETH Zürich (1997)
25. Viola, P., Wells III, W.M.: Alignment by maximization of mutual information. *International Journal of Computer Vision* 24(2), 137–154 (1997)

Detecting Motifs in a Large Data Set: Applying Probabilistic Insights to Motif Finding

Christina Boucher and Daniel G. Brown

David R. Cheriton School of Computer Science,
University of Waterloo, Waterloo, Ontario, Canada N2L 3G1
{cabouche,browndg}@cs.uwaterloo.ca

Abstract. We give a probabilistic algorithm for CONSENSUS SEQUENCE, a NP-complete subproblem of motif recognition, that can be described as follows: given set of l -length sequences, determine if there exists a sequence that has Hamming distance at most d from every sequence. We demonstrate that distance between a randomly selected majority sequence and a consensus sequence decreases as the size of the data set increases. Applying our probabilistic paradigms and insights to motif recognition we develop pMCL-WMR, a program capable of detecting motifs in large synthetic and real-genomic data sets. Our results show that detecting motifs in data sets increases in ease and efficiency when the size of set of sequence increases, a surprising and counter-intuitive fact that has significant impact on this deeply-investigated area.

1 Introduction

Given a number of DNA sequences, *motif recognition* is the task of discovering motif instances in sequences without prior knowledge of the consensus or their placement within the sequence. The following combinatorial formulation of motif recognition is due to Pevzner and Sze [17]: let $S = \{S_1, \dots, S_n\}$ be a set of DNA sequences each of length m , and M be the *consensus*, a fixed and unknown sequence of length l . Suppose M is a subsequence of S_i but is corrupted with at most d substitutions, so the Hamming distance to M is at most d . The aim is to determine M and the location of the motif instance in each sequence. This combinatorial problem has application to finding transcription factor binding sites in genomic data [22].

Motif recognition is NP-complete and thus, unlikely to be solved in polynomial time unless $P = NP$ [7]. Nonetheless, there are numerous algorithms developed to solve specific instances of the problem, including PROJECTION [3], Winner [17], pattern driven approaches [21], MITRA [6], PSM1 [18], PMSprune [10], the Voting algorithm [4], MCL-WMR [2] and several others. An important subproblem of motif recognition is formally described as follows:

CONSENSUS SEQUENCE

INSTANCE: a set of n sequences, $S = \{s_1, \dots, s_n\}$ over an alphabet Γ , each of length l , and a positive integer d .

FIND: a l -length sequence s^* over alphabet Γ where $H(s^*, s_i) \leq d$ for every s_i in S , or declare that no such s^* exists.

$H(s_i, s_j)$ denotes the Hamming distance for sequences s_i and s_j . Any sequence s where $H(s, s_i) \leq d$ for all s_i in S is referred to as a consensus; note that a consensus sequence need not be contained in S nor be unique. This problem is NP-complete, even when interest is restricted to the binary alphabet [9]. Li *et al.* [11] give a polynomial-time approximation scheme (PTAS) for CONSENSUS SEQUENCE, however, the unfortunate large degree in the computational complexity of the PTAS makes the algorithm inapplicability, though its sampling ideas are insightful.

We give a probabilistic algorithm for solving CONSENSUS SEQUENCE and demonstrate its application to motif recognition. Given a set of sequences over the alphabet Γ , the *majority symbol* for a specific position is the symbol in Γ that occurs most often at that position, with ties broken arbitrarily. We define the *majority sequence* for a given set of l -length sequences as the sequence of l majority symbols; due to ties the majority sequence is not necessarily unique. *MajorityConsensusAlg* begins with a majority sequence s , then successively updates s to make it closer to at least one sequence in S that does not have s as a consensus. We do a maximum number of kl cumulative updates to s , after this point if a consensus is not determined then the process is repeated. If a set of sequences S does not have a consensus then *MajorityConsensusAlg* will always return that no consensus exists but if it is a motif set then with some probability a consensus is returned – this probability is dependent upon the number of times we repeat the search.

We demonstrate that distance between a randomly selected majority sequence and a consensus sequence decreases significantly as the size of the data set increases. Our results show that for a large enough value of n the majority sequence is a consensus and hence, it is typically trivial to solve CONSENSUS SEQUENCE when the number of sequences is moderately large (for example, when $n \geq 12$ for $l = 15$ and $d = 5$).

We significantly extend our earlier motif recognition program, *MCL-WMR*, to incorporate *MajorityConsensusAlg*. This new algorithm, *pMCL-WMR*, detects motifs in data sets with a large number of sequences. More specifically, *pMCL-WMR* efficiently discovers motifs in data sets that have 30 or more sequences, and finds regulatory sequences in genomic data.

Lastly, we introduce the concept of a phase transition in CONSENSUS SEQUENCE. Such probabilistic analysis in random graph theory shows when a giant component is almost surely exists [1], and in theoretical computer science, the point at which a 3-SAT instance is likely to be satisfiable [16]. We analyze the event that a randomly selected majority sequence is a consensus and give evidence that this event exhibits a phase transition.

2 A Probabilistic Algorithm for CONSENSUS SEQUENCE

In [20], Schönig considers the following simple probabilistic algorithm for solving the NP-complete problem of k -SAT: randomly choose a starting assignment

and subsequently augment this initial assignment until a satisfying one is obtained. Papadimitriou introduced this random paradigm in the context of 2-SAT and obtained an expected quadratic time bound [14]. These type of algorithms are referred to as Monte Carlo algorithms with one-sided error. An useful property of such a Monte Carlo algorithm is that the error probability can be made arbitrarily small with repeated with independent random repetitions of the search process [13, page 9].

RandConsensusAlg begins with a sequence s randomly selected from all possible l -length sequences, and iteratively augments this sequence so it is closer (i.e. has smaller Hamming distance) to at least one sequence in S that does not have s as a consensus. This process of augmenting a random sequence until a consensus is found is repeated kl times; at this point the process is restarted if no consensus is found.

Algorithm 1 RandConsensusAlg

Input: A pairwise bounded set S of n l -length binary sequences, a degeneration parameter d .

Output: A consensus for S or the empty sequence if S is a decoy.

Let \mathcal{S} be the set of all l -length sequences

Repeat t times:

Select s randomly from \mathcal{S} .

Repeat kl times:

If s is a valid consensus for S then return s and terminate;

otherwise select $s_j \in S$ at random from all sequences where $H(s_j, s) > d$

Let \mathcal{L} be the set of positions where s and s_j mismatch.

Select a position from \mathcal{L} at random, and denote this position as r .

Update s such that s matches with s_j at position r .

A set of sequences S is *pairwise bounded* if for all sequences $a, b \in S$, $H(a, b) \leq 2d$. A set that is not pairwise bounded cannot have a consensus s so CONSENSUS SEQUENCE reduces to discerning between pairwise bounded sets that have a consensus (and if so, finding one such sequence) from those that do not. A set of sequences S is a *motif set* if it has a consensus sequence, and is a *decoy set* if S is pairwise bounded but does not have a consensus.

We consider a *uniquely satisfiable* set S , which we define as a set of sequences S with exactly one consensus, and denote s^* as the (unique) consensus for S . The process of augmenting a sequence a bounded number of times or until a consensus is found can be viewed as a Markov chain; the states of the Markov chain correspond to the Hamming distance between the current sequence we are augmenting and the consensus – if the random walk begins at state j then the terminating state 0 is reached in j steps by correctly choosing the j sites to augment. We apply the following observation of Schöning [20]: if i positions are chosen to be augmented that are actually part of the consensus (and hence, should not change) then in at most $2i + j$ augmentations a consensus is obtained; that is, i steps in “wrong” direction in the random walk and $i + j$ steps in the

correct direction. Hence, RandConsensusAlg considers a maximum of $2i + j \leq 3l$ augmentations to the randomly selected sequence.

Consider an instance S of CONSENSUS SEQUENCE, if S does not have a consensus then false will always be returned on each iteration of the algorithm. Assume S is uniquely satisfiable then with some probability, denoted as p , a consensus is determined. The expected number of times a random sequence is chosen and updated $3l$ times until a consensus is found is $1/p$, and the probability that a consensus is not found after t iterations is $(1 - p)^t$, which is bounded above by e^{-tp} . This will give us the appropriate value of t that corresponds to an associated error probability. Therefore, if Γ is the binary alphabet, S is a set of n , l -length sequences that is uniquely satisfiable by sequence s^* , and s is a random sequence then the number of augmentations of s required to reach s^* is within a polynomial factor of $(2(1 - \frac{1}{\Gamma}))^t$ [20].

We made the assumption that the set was uniquely satisfiable, however, this assumption is not needed – the random walk may find another consensus while not in the terminating state but this possibility only increases the probability the algorithm terminates. Further, similar results can be obtained for non-binary alphabet. For $|\Gamma| > 2$ and any $\epsilon > 0$ there exists an algorithm for CONSENSUS SEQUENCE that has running time $O\left((|\Gamma|(1 - \frac{1}{\Gamma}) + \epsilon)^t\right)$ [20]. Lastly, we note that taking into account that the starting (majority) sequence is not random, a stronger bound may be obtainable.

2.1 Substituting a Majority Sequence for a Random One

We consider a variant on RandConsensusAlg, referred to as MajorityConsensusAlg, that begins with a sequence chosen at random from the set of all majority sequences. The main difference between RandConsensusAlg and MajorityConsensusAlg is the choice of the starting sequence; we substitute the random choice of RandConsensusAlg for a sequence that has smaller Hamming distance to a consensus sequence. The polynomial-time approximation scheme (PTAS) due to Li *et al.* also uses the majority sequence but in a different manner; a majority sequence for a subset of r sequences is obtained, denoted as s , and is used to determine a set of n sequences that have closest distance to s . The majority sequence for this new set is obtained and it is determined whether this set is a motif using this new majority sequence [11].

We consider whether the substitution of the initial sequence will decrease the number of augmentations required to obtain a consensus from a computational perspective. We fixed l to be 15, d to be 4, and varied the number of sequences from 12 to 36. for each value of n , we randomly generated 100 motif sets and for each set determined the number of augmentations required to obtain a consensus from starting at a majority sequence and starting at a random sequence. Figure 1 illustrates our data and illustrates that the number of augmentations required to obtain a consensus starting from a random sequence is significantly larger, and as the number of sequences increases the disparity between the number of augmentations of a majority sequence and the number of augmentations of a

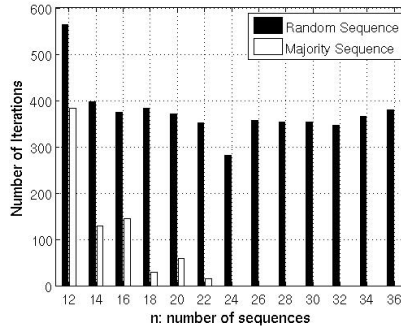


Fig. 1. A comparison of the average number of iterations required to obtain a consensus starting from a majority sequence (white) to the number required from starting from a random sequence (black). An average is taken from considering 100 randomly selected motif sets.

random sequence grows. When the number of sequences increases and a majority sequence is the initial sequence, the number of augmentations decreases; however, this trend does not exist when a random sequence is the initial sequence.

When the number of sequences is large enough, the distance between the selected majority sequence and a consensus is equal to zero – implying the majority sequence is a consensus. Figure 2 demonstrates that there exists a drastic decrease in the distance between the selected majority sequence and the consensus as n increases; namely, this point occurs in the empirical data when the value of n is around 12, when the ratio between l and d is 3. An important challenge is to explain this trend analytically.

The following result shows that as the number of sequence increases the probability that the majority vote sequence is a consensus increases. Corollary 1 is a direct result of the following theorem and demonstrates how it can be applied to determine exactly when the size of set of sequences is large enough to guarantee that the majority sequence is a consensus. Note that the results below concern motif sets; if the pairwise bounded set is a decoy then with probability 1 the majority sequence is not a consensus.

Theorem 1. *Assume the alphabet is binary. Let l and d remain constant. With probability at least $1 - \epsilon$ there exists a majority sequence that is a consensus for every motif set S of size at least $-\log(\epsilon/l) \cdot 2(\frac{l-d}{d})$.*

Corollary 1. *Assume the alphabet is binary, $l = 15$ and $d = 4$. With probability at least $1 - 10^{-10}$ there exists a majority sequence that is a consensus for every motif set S that contains at least 11 sequences.*

3 A Phase Transition for CONSENSUS SEQUENCE

A *phase transition* is a sharp change in the probability that a particular event occurs and thus, is defined with respect to a specific distribution of instances

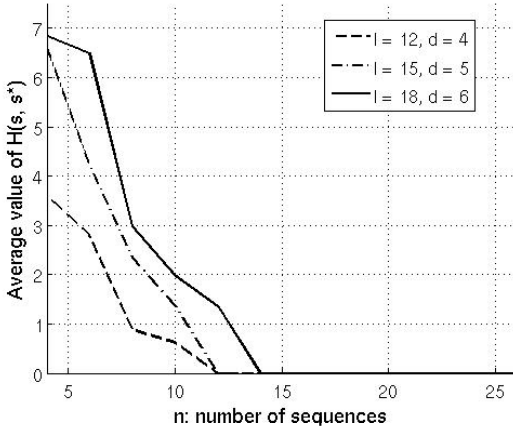


Fig. 2. An illustration of the distance between a randomly chosen majority sequence s and a consensus sequence found by MajorityConsensusAlg s^* with respect to n . We considered the following (l, d) pairs: $(12, 3)$, $(15, 5)$, and $(18, 6)$ and varied n to be every even value from 4 to 26; for each data point we repeated the experiment 100 times and took an average of $H(s, s^*)$.

of a particular problem. For example, for 3-SAT the critical point is the point where the probability that a legal, satisfying solution exists is $1/2$. It has been empirically found that the critical point for random 3-SAT with n variables and m clauses occurs when $m = 4.2n + 6.21$ [5]. This equation was later explained analytically by Pennock and Stout [16].

We illustrate a simple phase transition in CONSENSUS SEQUENCE. We aim to determine when the expected number of consensus sequences for a given motif set is greater than one. Assume we have a binary alphabet and without loss of generality, our motif set S has the identity sequence, denoted as $s^* = 0^l$, as a consensus. Clearly, each sequence in S has at most d positions not equal to 0. It follows that the probability that a sequence chosen at random is a consensus is a function of how many positions are not equal to 0; if the sequence is 0^l then the probability it is a consensus is 1, and if it has greater than $2d$ non-zeros then the probability it is a consensus is 0. Choose s_i to be any sequence in S and let $f(k)$ be the probability that s_i has a consensus with k nonzero positions. Let s_2^* be a randomly selected sequence from the set of all sequences with k nonzero positions. Let A be the subset of k positions that are 1 where s_2^* is equal to 1, and B be the subset of $l - k$ positions that are equal to 1 where s_2^* is equal to 0. If $k \leq d$ then we have that $f(k) = \sum_{A=0}^k \binom{k}{A} \sum_{B=0}^{d-A} \binom{l-k}{B} / \sum_{i=0}^d \binom{l}{i}$. If $d + 1 \leq k \leq 2(d - 1)$ then we have

$$f(k) = \frac{\sum_{A=k-d}^{k/2} \binom{k}{A} \sum_{B=0}^{d-k-A} \binom{l-k}{B}}{\sum_{i=0}^d \binom{l}{i}} + \frac{\sum_{A=k/2+1}^d \binom{k}{A} \sum_{B=0}^{d-A} \binom{l-k}{B}}{\sum_{i=0}^d \binom{l}{i}}$$

since $d - k - A \leq d - A$ when $2A \leq k$. Finally, we have for k equal to $2d - 1$ or $2d$, $f(k) = \sum_{A=k-d}^d \binom{k}{A} \sum_{B=0}^{d-A} \binom{l-k}{B} / \sum_{i=0}^d \binom{l}{i}$.

Each sequence in S is selected independently of the others and hence, the probability that s_2^* is a consensus is $f(k)^n$. Let N denote the expected number of consensus sequences then $N = \sum_{i=0}^{2d} \binom{l}{i} f(i)^n$, since $\binom{l}{i}$ is the number of sequences with exactly i positions not equal to zero. The change in phase occurs when there exists greater than one consensus for the set S (i.e. $N \geq 2$). Therefore, solving the equation $N \geq 2$, we obtain the location of the phase boundary:

$$\frac{-\log d}{\log \left(\sum_{A=0}^d \sum_{B=0}^{d-A} \left(\frac{2de}{A}\right)^A \left(\frac{(l-2d)e}{B}\right)^B \right)} \leq n.$$

We leave it open as to if there exists a stronger bound on n with respect to l and d . Next, we consider the expected Hamming distance between a randomly selected majority sequence and a consensus sequence. We prove there exists a critical value for n that dictates when the expected distance between a majority sequence and the closest consensus is 0. We conjecture that analysis of the structure of the space of satisfying assignments of a random motif will connect these two phase transitions, showing that if the data set is adequately large, basically all motif sets are clustered together and are characterized as having one unique consensus, which is the majority sequence.

Figure 2 suggests that the probability that a majority sequence is a consensus undergoes a phase transition when the ratio between the number of sequences and degeneracy parameter d passes a critical point; above this critical value we expect that there exists some consensus s^* , such that $H(s, s^*) = 0$ for all majority sequences s , almost surely. The following is an upper bound on the value of n and we leave it as an open problem to determine a tighter bound.

Theorem 2. *Let S be a uniquely satisfiable set of n, l -length binary sequences. Assume s is a randomly selected majority sequence and s^* is consensus then we consider the event that $E[H(s, s^*)]$ is equal to 0; an approximation to the critical point with respect to this event for n is $2 \left(1 - \frac{1}{1+\log d}\right)$.*

4 Experimental Results

We performed tests on a PC with a 64-bit 2600 MHz processor and 1 GB of RAM running Ubuntu. We measure the efficiency as the number of iterations required to solve a specific instance since this factor dictates both the wall and floor time and is an accurate measure of the algorithm’s complexity.

4.1 MajorityConsensusAlg

The number of sequences has a substantial effect on the running times for fixed values of l and d . Table 2 outlines the relationship between the value of n and the accuracy of the algorithm. The accuracy of the algorithm is the percentage of times in which the algorithm determined a consensus for the given set.

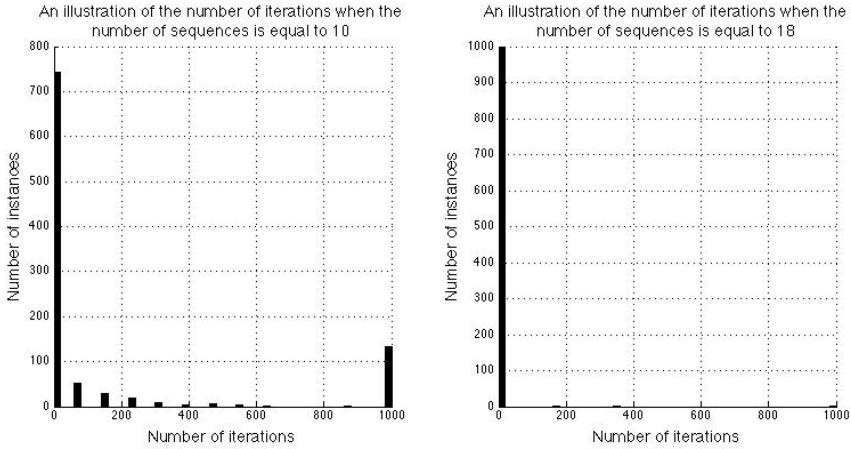


Fig. 3. An illustration of the effect of the value of n on the number of iterations required to find a consensus. Depicted is the data for instances with l equal to 10, d equal to 3, and n (the number of sequences) is equal to 10 and 18. The maximum number of updates to the majority sequence is set to 1000.

Table 1. Experimental data describing the changes in the accuracy and efficiency of MajorityConsensusAlg for varied values of the sequence length l , the degeneracy parameter d , and number of sequences. The accuracy and number of iterations is an average taken over the number of iterations and accuracy obtained for 1000 randomly selected sets of sequences representing a motif.

		$n = 5$		$n = 15$		$n = 20$	
l	d	Accuracy	Iterations	Accuracy	Iterations	Accuracy	Iterations
12	3	86 %	50424	100 %	0	100 %	0
14	4	63 %	181362	99 %	44097	100 %	0
15	4	69 %	174426	99 %	50627	100 %	0
18	6	61 %	316030	96 %	291618	96 %	518385
21	7	62 %	419188	93 %	694584	97 %	529187

Table 1 shows that when n was 20 the number of iterations required to find a consensus was 0 for the majority of values of l and d , implying that the majority sequence was a consensus. Table 2 further illustrates the change in accuracy and efficiency as the number of sequences increases. When l and d are equal to 10 and 3, respectively and the number of sequences is at least 18, all selected majority sequences were consensus sequences.

Figure 3 illustrates a significant difference in the number of iterations required by MajorityConsensusAlg when n is equal to 10 and n is equal to 18. The data is generated from MajorityConsensusAlg with the maximum number of updates to the majority sequence set to 1000 (we increased the value from $3l$ to 1000 to emphasize this dichotomy). When n is equal to 10, a significant portion of instances required the maximum number of iterations (and therefore, likely did not find a

Table 2. Data that illustrates the transition in the accuracy and efficiency when the number of sequences grows and the length of each sequence and amount of degeneracy remains fixed. This specific data gives the average accuracy and number of iterations for varied value of n and when l is equal to 10 and d is equal to 3.

n	Accuracy	Iterations	n	Accuracy	Iterations
4	57 %	41641	16	99 %	25608
6	86 %	54070	18	100 %	0
8	87 %	83204	20	100 %	0
10	92 %	80007	22	100 %	0
12	98 %	28807	24	100 %	0
14	98 %	39200	26	100 %	0

consensus), whereas when n is 18 the maximum number of iterations were made on only a very small number of instances, and a large portion of instances that took zero iterations (implying the majority sequence was a consensus).

Our data demonstrate that as the value of n increases the Hamming distance between the randomly selected majority sequence and a consensus decreases. All experimental results show for all values of l and d there exists a threshold value, say n_0 , for the number of sequences such that for instances where $n \geq n_0$ the determination of a consensus is trivial – the majority sequence is a consensus sequence.

4.2 pMCL-WMR: An Application of MajorityConsensusAlg

In 2007, MCL-WMR was developed specifically for the problem of detecting weak motifs in genetic data and works by first building an edge-weighted graph model of the given motif recognition problem and then using a graph clustering algorithm to quickly determine important subgraphs that need to be searched further for valid motifs [2]. These smaller subproblems are then solved optimality using a dynamic programming algorithm for finding motifs in dense subgraphs. One of the main contributions of the creation of MCL-WMR is the introduction of a novel model for motif recognition. Unfortunately, MCL-WMR was unable to detect motifs beyond when $l = 18$, $d = 6$, $m \geq 1000$, and $n \geq 20$ [2]. Further, Eskin and Pevzner reported similar results for various motif-finders [6] and in 2007, Feng *et al.* showed limited accuracy for the (15, 4) problem with 20 sequences of length 600 [8]. We investigate the application of MajorityConsensusAlg to motif recognition by replacing the dynamic programming algorithm used in MCL-WMR by MajorityConsensusAlg; pMCL-WMR is the resulting program. Specific motif recognition instances (i.e. specific values of n , m , l , and d) have remained intractable, such instances include when $l = 25$, $d = 5$, $m \geq 1000$, and $n \geq 20$.

Performance of pMCL-WMR on Synthetic Data. We produce problem instances as follows: we choose a random motif consensus of length l , and pick m occurrences of the motif by randomly choosing d positions per occurrence and

Table 3. Comparison of the performance of MCL-WMR and pMCL-WMR on synthetic data. The time is given in CPU seconds. In all experiments, $n = 1000$, $m = 20$, and l and d are varied “-” denotes that the program was unable to solve the specific problem due to CPU resources.

(l, d)	MCL-WMR	pMCL-WMR
(15, 4)	220	431
(16, 5)	12200	513
(18, 6)	20605	442
(25, 5)	-	771
(28, 8)	-	1020
(30, 9)	-	1106

Table 4. The performance of pMCL-WMR increasing a number of sequences. The time is given in CPU seconds. In all experiments, $l = 15$, $d = 4$, $m = 600$ and n ranges from 18 to 30.

n	pMCL-WMR
18	399
20	415
24	442
28	523
30	1223

randomly mutating the base at each. We construct m background sequences of length n and insert the generated motifs into a random position in the sequence. For each of the (l, d) combinations, 100 randomly generated sets of input sequences ($n = 20$, $m = 600$) were generated. Table 3 shows the comparison between the running time of MCL-WMR and that of pMCL-WMR. Two significant trends are witnessed in the data: pMCL-WMR is capable of solving very hard instances of motif recognition (i.e. when $l = 30$ and $d = 9$) and pMCL-WMR gives a dramatic improvement over MCL-WMR with respect to the running time for all values of l and d . The main advantage to our tool is the time required to solve the extremely difficult challenge problems—from the (18, 6) to the (30, 9) problem—in substantially better running time and with 100% accuracy.

The computational results of MajorityConsensusAlg inspire the investigation of solving instances with a significantly large number of sequences – that is, instances where n varies from values greater than 20. Table 4 shows the evaluation of the performance of MCL-WMR on a range of problems with an increasing number of sequences. The efficiency on these sets of problems is noteworthy ranging from 399 (when $n = 18$) to 1223 ($n = 30$), as far as we are aware of these are this first computational experiments where n is substantially large (i.e. in the range of 20 to 30).

Using pMCL-WMR to find regulatory elements. An important biological challenge is to determine regulatory elements in DNA – specifically, binding sites for transcription factors. In this section, we demonstrate the use of pMCL-WMR in discovering these DNA sequence “motifs” in data sets with a large number of DNA sequences. Tompa *et al.* extensively assess 13 motif recognition tools [22] using test sets that make use of transcription factor binding sites. The binding sites were obtained from the TRANSFAC database [23] and contains only eukaryotic transcription factors. The TRANSFAC database is extremely comprehensive, containing data from a large variety of species, i.e. species include yeast, mus, oryctolagus cuniculus, and homo sapiens [23]. For more details

concerning the data set, including the selection process for transcription factors and binding sites from TRANSFAC, see Tompa *et al.* [22].

Each transcription factor gives rise to one set of sequence. The number of sequences varied from 34 to 6 and the sequence length (parameter m) varied from 700bp to 2000bp. The transcription factor binding sites vary in length and thus, in order to assess pMCL-WMR, we ran the program on varied values l and d . The lengths of the motifs were same as those of the published motifs and d was varied. Experimental results are shown in Table 4.2. pMCL-WMR was capable of discovering motifs for these data sets, as well, as many motifs not yet found by the motif recognition programs assessed by Tompa *et al.* [22]. The known binding sites shown in Table 4.2, as given by the TRANSFAC database *et. al.* [22].

Data set	Published motif	Motif pattern discovered	l	d	Time (CPU sec.)
hm06	CAcgTG	TttTccC	7	1	0.85
hm06	CACCCGT	GGAcTGCT	8	1	1
hm10	TTTgcCGG	TTTcgCGC	8	3	1.23
hm13	AGGCTAGAGTAGA	aaAATTatTC	10	2	5.06
hm13	AAGATTATTAAC	tccCcCACAAa	11	1	3.43
hm19	GGTGGGGCGGGCGGGG	tccCcCACAAa	11	2	3.3

5 Conclusion

The data demonstrate that as the number of sequences increases, the number of iterations required to find a consensus decreases substantially, and show that solving CONSENSUS SEQUENCE is trivial when the data set is moderately large. Applying probabilistic methods, we prove a strong connection between the number of sequences in a given set S , and the ease in finding a consensus quickly. The implementation of these probabilistic algorithm to motif recognition enables the development of pMCL-WMR, a motif recognition program capable of quickly detecting motifs in large data sets.

Acknowledgements

We gratefully acknowledge research support of the National Sciences and Engineering Research Council of Canada.

References

1. Bollobas, B., Janson, S., Riordan, O.: The phase transition in inhomogeneous random graphs. *Random. Struct. Algor.* 31, 3–122 (2007)
2. Boucher, C., Brown, D., Church, P.: A graph clustering approach to weak motif recognition. In: Giancarlo, R., Hannenhalli, S. (eds.) *WABI 2007*. LNCS (LNBI), vol. 4645, pp. 149–160. Springer, Heidelberg (2007)

3. Buhler, J., Tompa, M.: Finding motifs using random projections. *J. Comput. Biol.* 9(3), 225–242 (2002)
4. Chin, F.Y.L., Leung, C.M.: Voting algorithms for discovering long motifs. In: *Proc. APBC 2005*, pp. 261–271 (2005)
5. Crawford, J.M., Auton, L.D.: Experimental results on the crossover point in satisfiability problems. In: *Proc. AAAI 1993*, pp. 21–27 (1993)
6. Eskin, E., Pevzner, P.A.: Finding composite regulatory patterns in DNA sequences. *Bioinformatics* 18(1), 354–363 (2002)
7. Evans, P.A., Smith, A., Wareham, H.T.: On the complexity of finding common approximate substrings. *Th. Comp. Sci.* 306, 407–430 (2003)
8. Feng, W., Wang, Z., Wang, L.: Identification of distinguishing motifs. In: Ma, B., Zhang, K. (eds.) *CPM 2007*. LNCS, vol. 4580, pp. 253–264. Springer, Heidelberg (2007)
9. Frances, M., Litman, A.: On covering problems of codes. *Th. Comp. Sys.* 30, 113–119 (1997)
10. Davila, J., Balla, S.: Rajasekaran. Fast and practical algorithms for planted (l, d) motif search. *IEEE/ACM Trans. Comput. Biol. Bioinf.* 4(4), 544–552 (2007)
11. Li, M., Ma, B., Wang, L.: Finding similar regions in many strings. *J. Comp. and Sys. Sci.* 65(1), 73–96 (2002)
12. Koutsoupias, E., Papadimitriou, C.H.: On the greedy algorithm for satisfiability. *Inform. Process. Lett.* 43, 53–55 (1992)
13. Motwani, R., Raghavan, R.: *Randomized Algorithms*. Cambridge University Press, New York (1995)
14. Papadimitriou, C.H.: On selecting a satisfying truth assignment. In: *Proc. FOCS 1991*, pp. 163–169 (1991)
15. Pavesi, G., Mauri, G., Pesole, G.: An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics* 17, S207–S214 (2001)
16. Pennock, D.M., Stout, Q.F.: Exploiting a theory of phase transitions in three-satisfiability problems. In: *Proc. AAAI 1996*, pp. 253–258 (1996)
17. Pevzner, P., Sze, S.: Combinatorial approaches to finding subtle signals in DNA sequences. In: *Proc. ISMB 2000*, pp. 344–354 (2000)
18. Rajasekaran, S., Balla, S., Huang, C.H.: Exact algorithms for the planted motif problem. *J. Comp. Bio.* 12(8), 1117–1128 (2005)
19. Sagot, M.-F.: Spelling approximate repeated or common motifs using a suffix tree. In: Lucchesi, C.L., Moura, A.V. (eds.) *LATIN 1998*. LNCS, vol. 1380, pp. 374–390. Springer, Heidelberg (1998)
20. Schöningh, U.: A probabilistic algorithm for k -SAT and constraint satisfaction problems. In: *Proc. FOCS 1999*, pp. 410–414 (1999)
21. Sze, S., Lu, S., Chen, J.: Integrating sample-driven and pattern-driven approaches in motif finding. In: Jonassen, I., Kim, J. (eds.) *WABI 2004*. LNCS (LNBI), vol. 3240, pp. 438–449. Springer, Heidelberg (2004)
22. Tompa, M., Li, N., Bailey, T.L., Church, G.M., De Moor, B., Eskin, E., Favorov, A.V., Frith, M.C., Fu, Y., Kent, W.J., Makeev, V.J., Mironov, A.A., Noble, W.S., Pavesi, G., Pesole, G., R egnier, M., Simonis, N., Sinha, S., Thijs, G., van Helden, J., Vandenbogaert, M., Weng, Z., Workman, C., Ye, C., Zhu, Z.: Assessing computational tools for the discovery of transcription factor binding sites. *Nat. Biotechnol.* 23, 137–144 (2005)
23. Wingender, E., Dietze, P., Karas, H., Kn uppel, R.: TRANSFAC: a database on transcription factors and their DNA binding sites. *Nucleic Acids Res.* 24(1), 238–241 (1996)

A Biclustering Method to Discover Co-regulated Genes Using Diverse Gene Expression Datasets*

Doruk Bozdağ^{1,2}, Jeffrey D. Parvin¹, and Umit V. Catalyurek^{1,2}

¹ Biomedical Informatics, The Ohio State University

² Electrical and Computer Engineering, The Ohio State University

{bozdagd, umit}@bmi.osu.edu, jeffrey.parvin@osumc.edu

Abstract. We propose a two-step biclustering approach to mine co-regulation patterns of a given reference gene to discover other genes that function in a common biological process. Currently, several successful methods utilize Pearson Correlation Coefficient (PCC) based gene expression analysis across all samples in datasets. However, microarray datasets are fraught with spurious samples or samples of diverse origin, and many genes/proteins that function in the same biological pathway may be missed. The novel PCC based biclustering algorithm introduced in this paper identifies subsets of genes with high correlation by stringently filtering the data and reducing false negatives due to spurious or unrelated samples in a dataset. Then, correlation information extracted from resulting biclusters are synthesized. We applied our method using the breast cancer associated tumor suppressors, BRCA1 and BRCA2, as the reference proteins to reveal genes and proteins important in the complex process of breast tumor formation. Experiments on 20 very large datasets showed that the top-ranked genes were remarkably enriched for genes that regulate the mitotic spindle and cytokinesis. The results imply that BRCA1 and BRCA2 proteins, which are considered to be DNA repair factors, have critical function regarding the mitotic spindle as well. Initial biological verification reveal that this identified factor function to control both centrosome dynamics, and also, surprisingly, DNA repair. Thus, this biclustering approach is successful at identifying proteins with highly related function from extremely complex datasets, and permits novel insights into gene function.

1 Introduction

Proteins that function in concert in a given cellular process often have their encoding mRNA co-expressed [1]. Therefore, examining transcription levels of genes under different conditions provides insight about functions of genes, and eventually development and treatment of complex diseases. DNA microarray technology has become the central enabling technology in genomic research by allowing measurement of expression levels of thousands of genes in parallel. In a microarray experiment, expression levels of genes in various samples are arranged in a matrix called *gene expression data*.

* This work was supported in parts by the U.S. DOE SciDAC Institute Grant #DE-FC02-06ER2775; the U.S. National Science Foundation Grants #CNS-0643969, #CCF-0342615, and #CNS-0426241, #CNS-0403342; Ohio Supercomputing Center Grant #PAS0052.

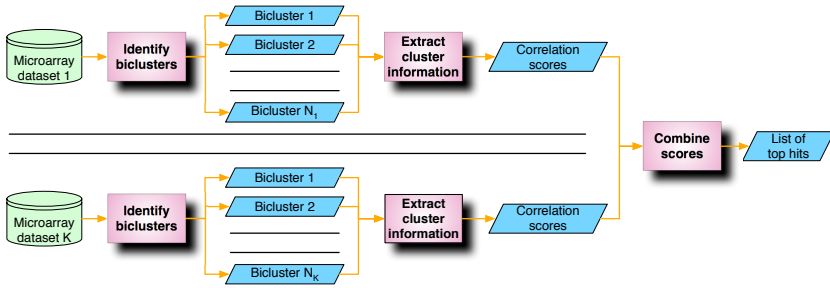


Fig. 1. Overview of the proposed approach

Samples are usually collected from different individuals and may correspond to different environmental conditions. Mining gene expression data to discover biologically relevant knowledge is a challenging task and has been the focus of many research efforts [2,3,4,5].

In this work, our objective is to develop a method that utilizes multiple gene expression datasets to identify genes exhibiting co-regulation with respect to a reference gene. Identifying genes co-regulated with a gene of important function is crucial to understand biochemical and genetic pathways in which the gene participates. A straightforward approach towards this aim is to cluster genes in each dataset using a correlation or similarity metric such as *Pearson Correlation Coefficient (PCC)* [6]; then count the number of times each gene co-occurs in the same cluster with the reference gene over all datasets. PCC is a very effective and widely used metric in this type of analysis to quantify co-regulation between pairs of genes [3,5].

A major drawback in this approach is that the entire set of samples in a dataset are used to decide cluster membership or correlation with the reference gene. Since samples are usually collected from diverse sources, genes and proteins that function together may only be similarly expressed in a subset of the samples. Moreover, most clustering techniques generate exclusive partitions of genes, therefore disregard the fact that a single gene may be involved in more than one biological pathway. To overcome these limitations we propose a new biclustering algorithm, called *Correlated Pattern Biclusters (CPB)*, that identifies groups of genes highly correlated with a given reference gene in empirically defined subsets of samples. We introduce novel techniques in CPB to address two important issues in biclustering of gene expression data: (1) mining datasets only to discover correlated patterns that contain the given reference gene, (2) extension of the use of PCC in biclustering context. In addition, CPB algorithm allows overlapping clusters and also captures negative correlation through use of PCC.

To reach our ultimate goal of identifying genes that consistently exhibit correlation with the reference gene, we also propose a method to extract correlation information from identified biclusters in an intuitive way. The proposed method evaluates uniqueness of information captured in each bicluster and computes a *correlation score* for each gene based on how frequently and in how distinct biclusters it co-occurs with the reference gene. Then, correlation scores from all datasets are combined to filter out inconsistent information. The overview of our approach is illustrated in Figure 1.

Our motivating application was from breast cancer research, where there are two important reference proteins, BRCA1 and BRCA2, highly penetrant breast cancer specific

Table 1. A sample dataset and biclusters identified by several methods from this dataset. (a) Sample matrix (b) Additive model (c) Multiplicative model (d) proposed CPB algorithm (e) OPSM.

1	2	3	8	95
2	3	4	9	21
5	6	7	12	51
2	4	6	16	18
3	6	9	24	30
15	14	13	8	7

(a)

1	2	3	8
2	3	4	9
5	6	7	12

(b)

1	2	3	8
2	4	6	16
3	6	9	24

(c)

1	2	3	8
2	3	4	9
5	6	7	12
2	4	6	16
3	6	9	24
15	14	13	8

(d)

1	2	3	8	95
2	3	4	9	21
5	6	7	12	51
2	4	6	16	18
3	6	9	24	30

(e)

tumor suppressors. Both of these proteins function in the repair of DNA damage. In addition, BRCA1 also functions at an organelle called centrosome, which is critical for cell division. To determine genes co-regulated with BRCA1 and BRCA2 we applied the method proposed in this paper on very large datasets publicly available at Gene Expression Omnibus (GEO) database [7]. The results are given in Section 5.

2 Background

Biclustering was first introduced to gene expression data analysis by Cheng and Church [8]. This is followed by numerous biclustering algorithms to identify additive, multiplicative [9,10], or even more complex relationships [2,11,12,13,14] between the rows and columns of a data matrix. In additive (multiplicative) models, the difference (ratio) between corresponding elements of any two rows and the difference (ratio) between corresponding elements of any two columns in a bicluster are constants. In general, additive models are useful to capture shifting patterns, whereas multiplicative models are useful to capture scaling patterns in the data. However, neither of them can identify shifting and scaling patterns simultaneously. Furthermore, these models are too restrictive in the sense that constant difference (ratio) constraints are applied on both row and column dimensions. In Table 1b and 1c, example biclusters that can be identified respectively by additive and multiplicative models from the sample matrix in Table 1a are shown.

In this work, we propose the CPB algorithm that utilizes statistical co-expression measure PCC as a similarity metric between rows of a bicluster. PCC is a strong metric to evaluate positive as well as negative co-regulation between rows, and is commonly used in clustering gene expression data [3,5] due to its power in capturing both shifting and scaling patterns. In Table 1d, an example bicluster identified by the CPB algorithm, where there is perfect correlation (or negative correlation) between each pair of rows is given. As shown in this figure, PCC allows capturing both shifting and scaling patterns that would be separately identified by additive and multiplicative models, respectively.

Application of PCC in biclustering context is not a trivial task and requires overcoming two challenges. Firstly, PCC lacks transitivity property. Therefore, instead of measuring closeness to a reference pattern, one has to compute all pairwise PCC values between rows in the same bicluster to measure quality. To tackle this problem, we empirically show that if two rows have a sufficiently high correlation with a reference

pattern, there is a lower bound for PCC between each these two rows. The second challenge is that, PCC is only meaningful to measure coherence between rows but is too restrictive if it is used to measure coherence between columns simultaneously. For instance, in the example in Table 1, if high PCC between each pair of columns was also enforced, only the biclusters that were identified by additive and multiplicative models would be found to match the ensuing criteria. In CPB algorithm, we enforce the coherence between columns by including a column in a bicluster only if it does not decrease correlation among the rows in the bicluster. To estimate the impact of including a column, we map columns to real numbers and capture tendency of gene expression changes in the bicluster. Then, we compute root mean squared error (RMSE) for each column to evaluate the fit of the column to this tendency pattern.

Mapping columns to real numbers induces an ordering of the columns similar to OPSM [2] and OP-cluster [11] algorithms. In a bicluster identified by OPSM algorithm, the direction of expression level change between any two columns is the same for all rows in the bicluster. OP-cluster is an extension to OPSM such that equivalence levels are defined to tolerate small differences between expression levels. Example of biclusters that would be identified by these algorithms for the example matrix in Table 1a is illustrated in Table 1e. In OPSM, the coherence between columns is defined in a more loose sense than CPB, and results in inclusion of a relatively less related column (column 5) in the bicluster. In addition to considering the direction of change, using PCC in CPB algorithm allows considering magnitude of change as well to eliminate inclusion of such columns. Moreover, it allows capturing negative correlation which is not handled by these algorithms. To the best of our knowledge, our work is the first work that uses PCC as an objective function for biclustering.

3 Correlated Pattern Biclusters Algorithm

Let R and C denote the set of rows and columns of a data matrix A , respectively, and each element a_{rc} represents the relation between row r and column c . A bicluster $B = (X, Y)$ can be defined by a subset of rows $X = \{x_1, \dots, x_n\}$ and a subset of columns $Y = \{y_1, \dots, y_m\}$, where $n \leq N$, and $m \leq M$ [4]. In our algorithm, we use PCC metric to decide membership of a row to a bicluster $B = (X, Y)$. We denote absolute value of PCC between rows $r, s \in R$ with respect to columns in Y by $pcc(r, s, Y)$. For a row r to be included in X , we require $pcc(r, x_i, Y)$ to be greater than a threshold for all $x_i \in X$. We also impose a constraint on the minimum size of Y to avoid getting large PCC values merely by chance. The objective of the proposed CPB algorithm can be formally defined as follows. Given a data matrix A , reference row r_r , PCC threshold ρ and minimum number of columns γ , identify a set of biclusters $B = (X, Y)$ such that $r_r \in X$, $m \geq \gamma$ and $pcc(x_i, x_j, Y) \geq \rho$ for all rows $x_i, x_j \in X$.

3.1 The Algorithm

Algorithm 1 outlines the proposed biclustering algorithm CPB. The algorithm starts with an initial bicluster $B = (X, Y)$ and improves it by iteratively moving rows and columns in and out of the bicluster using a search technique similar to mean-shift [15]. In mean-shift, the goal is to find the densest region with a certain radius (window size) in

Algorithm 1. Correlated Pattern Biclusters.

```

1: function CPB( $A, r_r, w, \gamma, \rho'$ )
2:    $step \leftarrow 1$ ;  $B = (X, Y)$  where  $X = \{r_r\}$  and  $Y$  is a random subset of columns of  $A$ .
3:   repeat ▷ Outer loop
4:      $B_{save} \leftarrow B$ ;  $\rho'_c \leftarrow 2/3\rho'$ ;  $\rho'_\Delta = 1/12\rho'$ ;  $\gamma = m$ ;  $\gamma_\Delta = \frac{m-\gamma}{4}$ 
5:     repeat
6:       Compute reference vector  $T$  and normalization parameters
7:       if  $step \bmod 2 = 1$  then
8:         Update  $X$  such that  $pcc(x_i, T, Y) > \rho'_c$  for all  $x_i \in X$ 
9:       else
10:        Let  $r$  be the row with smallest  $pcc(r, T, Y) > \rho'_c$ 
11:        Update  $Y$  such that  $RMSE(y_k) > RMSE(r)$  for all  $y_k \in Y$ 
12:         $\rho'_c \leftarrow \rho'_c + \rho'_\Delta$ ;  $\gamma_c \leftarrow \gamma_c - \gamma_\Delta$ 
13:      until  $\rho'_c > \rho'$ 
14:       $step \leftarrow step + 1$ 
15:   until  $step > 20$  or  $B = B_{save}$ 
16:   return  $B = (X, Y)$ 

```

the search space. At each iteration, the center of mass of the points that are at a distance smaller than the given radius to the center of the current solution is computed. Then, the center of the solution is moved to this computed center of mass and the process is repeated until convergence. Similarly in CPB algorithm, we compare PCC between each row and a reference vector $T = \langle t_1, \dots, t_m \rangle$ that represents general tendency of rows in X with respect to the columns in the bicluster while deciding which rows to move. Vector T is analogous to cluster center in k-means or mean-shift techniques. If $pcc(r, T, Y)$ for a row r is above a certain threshold, we include r into set X and update T by only considering the rows in X . On the other hand, using a similar criterion for columns is too restrictive for our objective as explained in Section 2. Instead, a good criterion for inclusion of a column c into Y should measure the impact of c on PCC between rows $x_i \in X$. For this purpose, we use *root mean squared error (RMSE)* to evaluate similarity of tendencies of rows in X with respect to column c .

In each iteration of CPB, first, reference vector T and parameters related to normalization of data values are computed; then, either set X or set Y are updated. We do not update both sets simultaneously to avoid large fluctuations in the bicluster structure, that may slow down or prevent convergence. In the spirit of the mean-shift technique, while updating X , we include into X each row r that has $pcc(r, T, Y)$ above PCC threshold ρ'_c . While updating Y , we first determine row r that has the smallest $pcc(r, T, Y)$ above threshold ρ'_c . Then we include each column c into Y that has smaller RMSE than row r . Iterations to update bicluster end when neither X nor Y changes at an iteration or after 20 iterations (convergence is usually achieved in 5-10 iterations). We use the CPB algorithm with different parameters and initializations to discover possibly overlapping clusters that contain rows correlated with the reference row.

3.2 Computing Normalization Parameters and the Reference Vector

In order to make tendency of rows in X comparable, we apply normalization to account for different scaling and shifting patterns of rows in the bicluster. We compute a

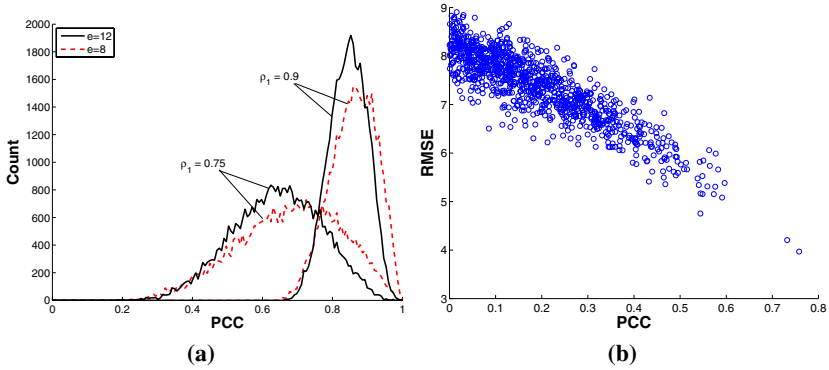


Fig. 2. (a) Distribution of PCC between pairs of 200 random vectors with e elements that have PCC with reference vector greater than a threshold ρ_1 . (b) Relationship between PCC and RMSE on random vectors.

normalized data value $\hat{a}_{x_i y_k} = \frac{a_{x_i y_k} - \alpha_{x_i}}{\beta_{x_i}}$ for each $x_i \in X$ and $y_k \in Y$, where α_{x_i} and β_{x_i} are shifting and scaling parameters associated with row x_i , respectively. Then, each element t_k of reference vector T is computed as the arithmetic mean of $\hat{a}_{x_i y_k}$ on all rows $x_i \in X$. We compute T , α_{x_i} and β_{x_i} using an iterative process. Initially we set $\alpha_{x_i} = 0$ and $\beta_{x_i} = 1$, and compute T . Then, we apply least squares fitting on pairs $\{(t_1, a_{x_i y_1}), \dots, (t_m, a_{x_i y_m})\}$ to obtain the best shifting and scaling parameters that maximize alignment of each row x_i with the reference vector T . We assign intercept and slope obtained in least squares fitting to α_{x_i} and β_{x_i} , respectively. T is updated using these parameters, and the process iterates until convergence.

3.3 Updating Rows of a Bicluster

For a row r to be a member of set X , we require $pcc(r, x_i, Y) > \rho$ for all $x_i \in X$. To avoid testing this condition against all $x_i \in X$, we utilize the reference vector T , and only test whether $pcc(r, T, Y)$ is greater than another threshold ρ' instead. ρ' is selected such that $pcc(r, T, Y) > \rho'$ must ensure $pcc(r, x_i, Y) > \rho$ for all $x_i \in X$. However, PCC lacks transitivity property [16] and has a fairly complex formula that strongly depends on the values and the length of the vectors. Therefore, it is difficult, if not impossible, to analytically compute a lower bound for ρ' as a function of ρ . To empirically determine the value of ρ' for a given ρ , we designed a simple experiment. First, we generated a reference random vector with e elements. Then we generated more random vectors and kept only those having absolute value of PCC with the reference vector greater than ρ' . After generating 200 such vectors we plotted the distribution of the absolute value of PCC between each pair of these vectors (see Figure 2a). The distributions verify that a lower bound for ρ' exists and increases with ρ .

In Algorithm 1, we start with a relaxed threshold ρ' and slowly tighten it at Line 12. While tightening ρ' , we relax the constraint on minimum number of columns. This allows sweeping the search space between two extreme combinations of these parameters. In our code we use 5 tightening steps and initial values for ρ'_c and γ_c are set to $2/3\rho'$, and the number of columns in the initial bicluster, respectively (Line 4).

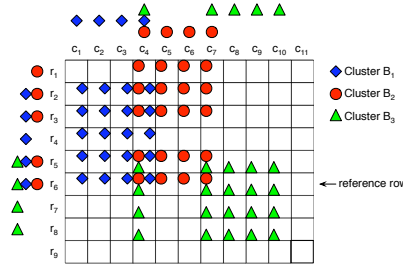


Fig. 3. Example biclusters on an example data matrix with reference row $r_r = r_6$

3.4 Updating Columns of a Bicluster

We use $RMSE$ to assess coherence of tendencies of rows $x_i \in X$ in a given column. $RMSE(y_k)$ for a column $y_k \in Y$ is computed as $\sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{a}_{x_i y_k} - t_k)^2}$. For a column $c \notin Y$, we compute $RMSE(c)$ in a similar way, by using a value t_c analogous to t_k that quantifies tendency of rows $x_i \in X$ in column c .

In CPB, only the columns having $RMSE$ below a threshold ϵ are included in the bicluster. In order to have control on the ratio of the number of rows to the number of columns in the bicluster, we select ϵ in relation to ρ' . To establish this relation, first we note that $RMSE$ can also be computed for rows, and it is a comparable metric for rows and columns. For a row $x_i \in X$, $RMSE(x_i)$ is computed as $\sqrt{\frac{1}{m} \sum_{k=1}^m (\hat{a}_{x_i y_k} - t_k)^2}$. Then, we observe that $RMSE(r)$ generally implies a high $pcc(r, T, Y)$ (see Figure 2b). Therefore, by setting ϵ to the $RMSE$ of row r that has the smallest $pcc(r, T, Y)$ above threshold ρ'_c (Line 10), we expect that the ratio n/m in the resulting bicluster is close to the ratio N/M . In order to obtain biclusters with different n/m ratios, we use parameter κ . Then, when updating set Y , κ times the number of columns with $RMSE$ above the threshold are included into Y .

To ensure that the reference row r_r has a larger impact in decision mechanisms of the algorithm, we assign a larger weight to the reference row when computing the vector T and $RMSE$ values. Total contribution from rows except r_r is multiplied by $(1 - \omega)$ and contribution from r_r is multiplied by ω , where ω is an input parameter. Large values for ω allows discovering patterns that more closely resemble r_r ; whereas small values increase sensitivity, hence offers higher tolerance to noise.

4 Combining Correlation Information

In this section, we explain our method to extract correlation information from identified biclusters. For this purpose, first we quantify uniqueness of information captured by each bicluster. Then, for each row we compute a *correlation score* based on co-occurrence frequency and uniqueness information associated with the row with respect to the reference row. Finally, we combine correlation scores from different datasets.

If two biclusters $B_v = (X_v, Y_v)$ and $B_w = (X_w, Y_w)$ do not overlap except for the reference row r_r , then these two biclusters represent two distinct relationships between

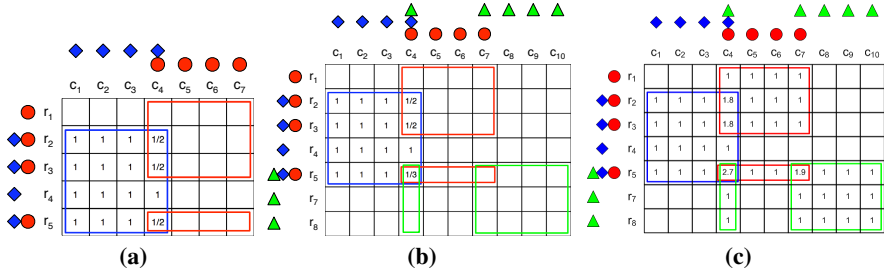


Fig. 4. $1/|IR(r, x_i) \cap IC(c, y_k)|$ values for each $x_i \in X_1$ and $y_k \in Y_1$ for (a) $r = r_2, c = c_4$, (b) $r = r_5, c = c_4$. (c) $OS(r, c)$ for each row r and column c .

rows and columns of the data matrix. In the context of gene expression, this may correspond to two different biological functions associated with the reference gene. On the other hand, if $X_w \subseteq X_v$ and $Y_w \subseteq Y_v$ the relationship in B_w is already captured by B_v . In the latter case we discard B_w from the result set.

Let $IR(\dots)$ denote the set of biclusters that contain all rows specified in the argument list. Similarly, let $IC(\dots)$ denote the set of biclusters that contain all columns specified in the argument list. Consider a row r , a column c and a bicluster $B_v = (X_v, Y_v)$ such that $r \in X_v$ and $c \in Y_v$. To measure uniqueness of information in B_v with respect to other biclusters in set $IR(r) \cap IC(c)$ on the relationship between r and c , we define a *bicluster uniqueness* measure $BU(B_v, r, c)$ as follows.

$$BU(B_v, r, c) = \frac{\sum_{x_i \in X_v - \{r_r\}} \sum_{y_k \in Y_v} \frac{1}{|IR(r, x_i) \cap IC(c, y_k)|}}{(|X_v| - 1)|Y_v|} \quad (1)$$

If B_v does not overlap with any other bicluster at row r and column c , then $IR(r, x_i) \cap IC(c, y_k)$ only contains B_v for all $x_i \in X$ and $y_k \in Y_v$ in (1). In this case $BU(B_v, r, c)$ takes its maximum possible value of 1. This means that B_v captures the relationship between row r and column c exclusively. $BU(B_v, r, c)$ decreases as overlap between B_v and biclusters in $IR(r) \cap IC(c)$ increases. In the case that B_v completely overlaps with all clusters in $IR(r) \cap IC(c)$, information on the relationship between row r and column c is shared between all of these clusters. Then $BU(B_v, r, c)$ takes its minimum value of $1/|IR(r) \cap IC(c)|$ (note that this case is not actually possible since we remove biclusters that are subsets of other biclusters beforehand). Computing cluster uniqueness as given in (1) is useful to avoid some relationships to be over-emphasized due to convergence of biclustering algorithm to solutions close to each other in the search space.

An example matrix and three biclusters are shown in Figure 3. Consider bicluster $B_1 = (X_1, Y_1)$ and row r_2 and column c_4 of the matrix. Since $IR(r_2) \cap IC(c_4) = \{B_1, B_2\}$, overlaps between B_1 and B_2 need to be considered when computing $BU(B_1, r_2, c_4)$. Figure 4a shows values $1/|IR(r_2, x_i) \cap IC(c_4, y_k)|$ for each $x_i \in X_1$ and $y_k \in Y_1$. Applying these values to (1) gives $BU(B_1, r_2, c_4) = 0.91$. Corresponding values to compute $BU(B_1, r_5, c_4)$ are given in Figure 4b. Here $IR(r_5) \cap IC(c_4) = \{B_1, B_2, B_3\}$, thus $BU(B_1, r_5, c_4) = 0.9$.

Using bicluster uniqueness measure, we compute an *overlap score* $OS(r, c)$ for every row-column pair (r, c) to quantify the amount of different relationships identified between r and c . We compute $OS(r, c)$ by summing $BU(B_v, r, c)$ for all biclusters in $IR(r) \cap IC(c)$. In other words, $OS(r, c) = \sum_{B_v \in IR(r) \cap IC(c)} BU(B_v, r, c)$. Then, we compute a *correlation score* $CS(r)$ for each row r by summing overlap scores of the pairs (r, c) across all columns, i.e. $CS(r) = \sum_{c \in C} OS(r, c)$. Summing overlap scores across columns gathers total evidence on how frequently and in how distinct relationships row r is correlated with the reference row r_r .

In Figure 4c, $OS(r, c)$ is given for pair (r, c) . Summing these values across columns gives $CS(r_1) = CS(r_4) = 4$, $CS(r_7) = CS(r_8) = 5$, $CS(r_2) = CS(r_3) = 7.8$ and $CS(r_5) = 12.6$. As expected, rows that appear in larger number of biclusters and in more diverse relationships together with the reference row have larger correlation score.

To increase significance and consistency of our findings, we apply our method on different datasets separately and combine correlation scores. To achieve this in a meaningful way, we require datasets to have the same row labels. In gene expression data analysis, this requirement can be met by merging results only from datasets obtained using the same microarray chip. Even though such datasets could be combined into a single data matrix, this approach requires undoing any normalization previously carried out on each dataset. Since data are collected from different sources, this approach may not be practical or even possible if information about the normalization procedures are unavailable. As an alternative approach, we use the following three-step method: First, for each dataset, we divide correlation score of each row by that of the reference row in the same dataset. Then, in order to make contribution from each dataset equal, we scale correlation scores such that sum of the scores in each dataset is the same. Finally, we sum the scaled scores across datasets to compute a total score for each row.

5 Experimental Results

5.1 Experiments on Synthetic Data

To demonstrate the effectiveness of CPB, we generated datasets with embedded biclusters and applied CPB to find these biclusters. We first generated a 10000×100 matrix and a reference row vector of length m , filled with random real numbers between 0 and 100. Then, we generated $n - 1$ additional vectors, each having perfect positive or negative correlation with the reference vector. These vectors together represent an $n \times m$ bicluster. Next, we added a random number between 0 and K chosen from normal distribution to each entry in the bicluster to simulate noise in the data. Finally, we embedded the bicluster into randomly selected n rows and m columns of the dataset.

As with most clustering algorithms, there is no single set of parameter values of CPB that will suit to all datasets. Therefore, when using CPB, we consider a range of values for each parameter to scan the search space thoroughly. In our experiments on synthetic datasets we generated 10 datasets for every combination of $n = \{30, 60, 90, 120, 150\}$, $m = \{30, 60, 90\}$ and $K = \{0, 1, 2\}$. First, we applied the CPB algorithm with row column ratio parameter $\kappa = 1$, $\rho' = 0.9$ and relative weight ω of reference gene selected from $\{0.1, 0.25, 0.5, 0.75\}$. For each value of ω we applied CPB 21 times

Table 2. Datasets we used in our experiments from GEO [7] database

GDS dataset ID	534	596	715	1067	1209	1220	1284	1375	1479	1615
Number of samples	75	158	87	52	54	54	50	70	60	127
GDS dataset ID	1781	1815	1956	1975	2113	2190	2255	2362	2373	2643
Number of samples	104	100	121	85	76	61	58	71	130	56

Table 3. Intersection of top-25 lists of BRCA1 and BRCA2 reference probe sets

Affymetrix probe set ID	Associated protein	Affymetrix probe set ID	Associated protein
201292_at	TOP2A	210052_s_at	TPX2
202095_s_at	BIRC5	214710_s_at	CCNB1
202705_at	CCNB2	218009_s_at	PRC1
204962_s_at	CENPA	218039_at	NUSAP1
209642_at	BUB1	218355_at	KIF4A

using different initial clusters. The value of threshold ρ corresponding to $\rho' = 0.9$ was 0.65. This value is obtained by the method explained in Section 3. We consider an embedded bicluster identified, if the returned bicluster consists of at least half of the rows and half of the columns of the embedded bicluster. If all rows and columns of the embedded bicluster are returned, we call the bicluster perfectly identified. When there was no noise in the data, CPB algorithm perfectly identified 148 of the 150 embedded biclusters. When some noise is added, the bicluster structure is more difficult to discover due to reduced PCC values between the rows. Furthermore, it is likely that some of the rows will no longer have PCC above 0.9 with the reference row. In our experiments with $K = 1$, CPB was able to identify 145 of the 150 biclusters. Of these, 140 were perfectly identified, and the for the remaining ones, all rows and at least 90% of the columns were returned by the CPB algorithm. Finally, when K was 2, there was a more pronounced impact of noise resulting in much reduced PCC between the rows. Still, CPB algorithm successfully identified 131 of the 150 embedded biclusters. For 91 of these biclusters, CPB returned at least two thirds of the columns and two thirds of the rows.

Next, we applied a PCC based clustering approach to identify rows having PCC greater than 0.65 with the reference row over all columns. Even in the best case, at most 27% of the columns in a bicluster were successfully identified by this approach. This shows that considering all columns to compute PCC prevents detection of biclusters.

5.2 Identifying Genes Co-regulated with BRCA1 and BRCA2

For real data experiments we selected 20 large datasets each obtained using Affymetrix HG U133 GeneChip Array and having at least 50 samples (Table 2). This array has 22,215 probe sets including two probe sets for each of BRCA1 (204531_s_at, 211851_x_at) and BRCA2 (208368_s_at, 214727_at). For each run of CPB, κ is selected from $\{1, 3, 5, 7, 9\}$; ω from $\{0.25, 0.5, 0.75\}$; and ρ' was set to 0.9. We executed the algorithm for every combination of these values, and for each parameter set

Table 4. GO term enrichment results using 90 genes obtained by intersecting top-500 lists of reference genes. n and N represent the number of genes associated with the GO term in the set of identified genes and in the Affymetrix chip, respectively.

GO term ID	GO term	p-value	n	N
GO:000910	cytokinesis	$< 1.0 \times 10^{-12}$	8	40
GO:0007049	cell cycle	$< 1.0 \times 10^{-12}$	38	720
GO:0007067	mitosis	$< 1.0 \times 10^{-12}$	34	205
GO:0031577	spindle checkpoint	$< 1.0 \times 10^{-12}$	3	3
GO:0040001	establishment of mitotic spindle localization	$< 1.0 \times 10^{-12}$	4	4
GO:0045842	positive regulation of mitotic metaphase/anaphase transition	$< 1.0 \times 10^{-12}$	3	3
GO:0051301	cell division	$< 1.0 \times 10^{-12}$	29	266
GO:0051303	establishment of chromosome localization	$< 1.0 \times 10^{-12}$	3	3
GO:0007051	spindle organization and biogenesis	2.9×10^{-12}	5	9
GO:0031503	protein complex localization	7.6×10^{-12}	6	8
GO:0031536	positive regulation of exit from mitosis	1.0×10^{-11}	4	7

we generated 21 random initial clusters. We applied the analysis four times using one of the BRCA1 or BRCA2 probe sets as the reference each time. In Table 3, we present genes that appeared in top-25 highest correlated gene list of each of the four reference probe sets.

There are 90 genes that were common in top-500 list for all four reference probe sets. Analysis of Gene Ontology (GO) terms associated with these 90 genes statistically supports the extraordinary clustering of proteins that function in mitosis. The top-ranked genes are remarkably enriched for genes that regulate the mitotic spindle and cytokinesis. As given in Table 4, of these 90 genes, 38 control the cell cycle, 34 relate to mitosis and 29 involved in cell division. The enrichment of cell cycle, mitosis, and cellular assembly are exactly what would be predicted for control by the centrosome. DNA replication and repair would be predicted to be a part of the BRCA1 and BRCA2 module, and this pathway would also impact the centrosome.

The results show that our algorithm is successful at identifying from extremely complex datasets proteins with highly related function. While our results did reveal known factors for the repair of DNA damage as expected, the most significant results were enriched for centrosome and mitotic spindle related processes. This implies that BRCA1 and BRCA2, which are considered to be DNA repair factors, also have critical function regarding the mitotic spindle. Biological testing of this point is in progress, but in initial tests a gene of unknown function identified by our method is found to control centrosome¹. If confirmed, this will imply that control of the mitotic spindle is a critical control element in breast cancer. In addition, several of the identified proteins that function to control the centrosome were found to also control a DNA repair assay. This was an unanticipated finding. Thus, biological validation in progress is revealing that this biclustering tool both reveals proteins that function together to control centrosomes and also to participate in a second process of DNA damage repair.

We have also tested this method for false positives by applying the biclustering tool on seven more genes to verify that our method avoids systematic errors. Two of the

¹ A recent work of J. D. Parvin, unpublished.

genes we used for this analysis are RB1 and TP53, tumor suppressor genes involved in many cancers. The other five genes were CCNB2, FGD2, TAF7, SRP54 and CHPF, which were ranked 1st, 5000th, 10000th, 15000th and 20000th, respectively, when correlation scores of four reference probe sets are combined. We used each of these selected genes as anchors and applied our analysis to determine top-25 lists for high correlation for each of these genes. This analysis verified that BRCA1, BRCA2, as well as the number one hit CCNB2 are correlated with a similar set of genes. For each pair of these genes there were 16 to 20 genes at the intersection of top-25 lists. On the other hand, the genes we selected for verification had at most one gene in common in the top-25 list of either of BRCA1, BRCA2 or CCNB2.

6 Conclusion and Future Work

In this work, we proposed a two-step approach to mine co-regulation patterns, relative to a set of reference genes, that may only exist in a subset of samples. First, co-regulation patterns in microarray datasets are discovered using a novel PCC-based biclustering algorithm. Then, correlation information is combined to compute a correlation score with respect to the reference gene. In our experiments we used BRCA1 and BRCA2 as our reference genes. Analysis of the top-ranked genes using GO terms revealed an extraordinary clustering of proteins that function in mitosis. In the future, we plan to compare the CPB algorithm with other biclustering algorithms in terms of both objective functions and optimization techniques. Furthermore, we will evaluate significance of our findings by testing the algorithm on various real datasets.

References

1. Agrawal, H.: Extreme self-organization in networks constructed from gene expression data. *Phys. Rev. Lett.* 89(26), 268702 (2002)
2. Ben-Dor, A., Chor, B., Karp, R., Yakhini, Z.: Discovering local structure in gene expression data: The order-preserving submatrix problem. *Int'l Conf. Comput. Biol.*, 49–57 (2002)
3. Jiang, D., Pei, J., Zhang, A.: DHC: A density-based hierarchical clustering method for time series gene expression data. In: *IEEE Symp. Bioinform. and Bioeng.*, pp. 393–400 (2003)
4. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biology Bioinform.* 1(1), 24–45 (2004)
5. Pujana, M.A., et al.: Network modeling links breast cancer susceptibility and centrosome dysfunction. *Nature Genetics* 39(11), 1338–1349 (2007)
6. Devore, J.L.: *Probability and Statistics for Engineering and Sciences*. Brook/Cole Publishing Company (1991)
7. Edgar, R., Domrachev, M., Lash, A.E.: Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucl. Acids Res.* 30(1), 207–210 (2002)
8. Cheng, Y., Church, G.M.: Biclustering of expression data. *Int'l Conf. Intelligent Systems for Molecular Biology*, 93–103 (2000)
9. Segal, E., Taskar, B., Gasch, A., Friedman, N., Koller, D.: Rich probabilistic models for gene expression. *Bioinformatics* 17(suppl. 1), S243–S252 (2001)
10. Wang, H., Wang, W., Yang, J., Yu, P.S.: Clustering by pattern similarity in large data sets. In: *ACM SIGMOD* (2002)

11. Liu, J., Wang, W.: Op-cluster: Clustering by tendency in high dimensional space. In: IEEE Int'l. Conf. Data Mining, p. 187 (2003)
12. Murali, T., Kasif, S.: Extracting conserved gene expression motifs from gene expression data. In: Pac. Symp. Biocomp., vol. 8 (2003)
13. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. *Bioinformatics* 18(suppl. 1), 136–144 (2002)
14. Kluger, Y., Basri, R., Chang, J.T., Gerstein, M.: Spectral biclustering of microarray data: coclustering genes and conditions. *Genome Res.* 13(4), 703–716 (2003)
15. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence* 24(5), 603–619 (2002)
16. Casella, G., Wells, M.T.: Is Pitman closeness a reasonable criterion?: Comment. *Journal of the American Statistical Association* 88(421), 70–71 (1993)

Computational Protocol for Screening GPI-anchored Proteins

Wei Cao¹, Kazuya Sumikoshi¹, Tohru Terada², Shugo Nakamura¹,
Katsuhiko Kitamoto¹, and Kentaro Shimizu¹

¹ Department of Biotechnology, Graduate School of Agricultural and Life Sciences,
University of Tokyo, 1-1-1 Yayoi, Bunkyo-ku, Tokyo 113-8657, Japan

² Professional Programme for Agricultural Bioinformatics, Graduate School of
Agricultural and Life Sciences, University of Tokyo, 1-1-1 Yayoi, Bunkyo-ku, Tokyo
113-8657, Japan

Abstract. Glycosylphosphatidylinositol (GPI) lipid modification is an important protein posttranslational modification found in many organisms, and GPI-anchoring is confined to the C-terminus of the target protein. We have developed a novel computational protocol for identifying GPI-anchored proteins, which is more accurate than previously proposed protocols. It uses an optimized support vector machine (SVM) classifier to recognize the C-terminal sequence pattern and uses a voting system based on SignalP version 3.0 to determine the presence or absence of the N-terminal signal of a typical GPI-anchored protein. The SVM classifier shows an accuracy of 96%, and the area under the receiver operating characteristic (ROC) curve is 0.97 under a 5-fold cross-validation test. Fourteen of 15 proteins in our sensitivity test dataset and 19 of the 20 proteins experimentally identified by Hamada et al. that were not included in the training dataset were identified correctly. This suggests that our protocol is considerably effective on unseen data. A proteome-wide survey applying the protocol to *S. cerevisiae* identified 88 proteins as putative GPI-anchored proteins.

Keywords: GPI-anchored Proteins; SVM; Post-translational modification.

1 Introduction

Glypiation is an important posttranslational modification in which glycosylphosphatidylinositol (GPI) is attached to newly synthesized proteins so that they can be bound to the plasma membrane. GPI-anchored proteins are found on all eukaryotic cells and serve as adhesion molecules [1], enzymes [2] and receptors [3]. They are also related to the pathogenesis of neurodegenerative diseases such as Scrapie [4] and Creutzfeld-Jacob diseases [5]. Previous studies [6,7] have shown that a newly synthesized protein destined to receive a GPI anchor has two signal peptide sequences: a C-terminal sequence required for anchor attachment and an N-terminal sequence required for transferring the protein into endoplasmic reticulum (ER).

Experimental investigation of GPI-anchored proteins on the proteomic scale is a formidable analytical challenge because of the explosion of proteomic data. Computational approaches will therefore be valuable for dealing with the challenge and providing critical information rapidly, and several research groups have reported their works in this area. Caro et al. [8] firstly analyzed the yeast proteome with SignalP-NN [9], which is a classifier trained with a Neural Network technique for the N-terminal signal prediction, and the consensus rules for the C-terminal signal prediction, which was described by Nuoffer et al. [10] and by Udenfriend and Kodukula [7] for screening GPI-anchored proteins. As a result, they found 58 potential GPI-anchored proteins in the yeast proteome. De Groot et al. [11] screened the proteomes of *S. cerevisiae*, *C. albicans*, *Sz. Pombe* and *N. crassa* for potential GPI-anchored proteins by using a sequence pattern derived from known GPI-anchored proteins of *S. cerevisiae* and *C. albicans*, [NSGDAC]-[GASVIETKDLF]-[GASV]-X(4,19)-[FILMVAGPSTCYWN](10)>, (where ">" marks the C-terminus). They then confirmed the presence of the N-terminal signal sequence by combining the outputs of the SignalP-NN and the SignalP-HMM [12]. After then using PSORT II [13] and TMHMM (available at <http://www.cbs.dtu.dk/services/TMHMM/>) to eliminate internal transmembrane domains, they finally identified 66 putative proteins for *S. cerevisiae*, 104 for *C. albicans*, 33 for *Sz. Pombe* and 97 for *N. crassa*. Eisenhaber et al. [14] developed a method for identifying the C-terminal signal. Using their method, called Big-PI and incorporating the SignalP (SignalP-NN) for the N-terminal signal identification, they found 74 GPI-anchored proteins in the proteome of *A. nidulans*. They analyzed the proteome of *A. nidulans*, 59 in the proteome of *S. cerevisiae*, 169 in the proteome of *C. albicans*, 28 in the proteome of *Sz. Pombe*, and 87 in the proteome of *N. crassa*. Fankhauser and Mäser developed a classifier, called GPI-SOM, which were trained with a Kohonen Self-Organizing Map for the C-terminal signal prediction [15]. Combining their GPI-SOM with SignalP-HMM, they investigated GPI-anchored proteins in ten proteomes. Methodologically, the proposed protocols for identifying GPI-anchored proteins differ mainly in the method for examining the C-terminal signal. An experimental study in which decay-accelerating factor was found to be GPI-anchored even after the C-terminal signal had been replaced with another hydrophobic sequence has revealed that the overall hydrophobicity of the C-terminal signal is more important than the sequence specificity [16]. Therefore, the method based on a consensus rule or pattern search is insufficient for identifying GPI-anchored proteins since amino acid type preferences are not obvious [17]. Big-PI suffers from ambiguous predictions (i.e., twilight zone) and was pointed out by Fankhauser and Mäser that Big-PI is difficult to balance false positive and false negative errors.

The support vector machine introduced by Vapnik and his coworkers [18,19] is known for its outstanding performance in classification and has been successfully applied in many issues of computational biology [20,21,22]. In this paper we describe a new SVM-based computational protocol for screening GPI-anchored proteins on the proteomic scale and report its application on the proteome of *S. cerevisiae*.

2 Materials and Methods

2.1 Computational Protocol for Identifying GPI-anchored Proteins

Our protocol is composed of two modules (see Fig. 1). One is a SVM classifier (here denoted GPI-SVM) screening for the C-terminal signal and the other is SignalP-Vote screening for the presence of the N-terminal export signal. A protein is considered to be a potential GPI-anchored protein only if both signals are identified. For clarity in this paper, we use the naming style "C"+"N", where "C" is the name of C-terminal prediction method, "N" is the name of N-terminal prediction method, and "+" denotes their combination. Our new prediction protocol is thus denoted GPI-SVM+SignalP-Vote.

The C-terminal signal prediction of GPI-anchored proteins is formatted into a two-class SVM classification task: i.e., training the GPI-SVM classifier and, for a given query protein sequence, using it to answer whether or not the protein has a potential C-terminal signal. We first collected an approximately balanced dataset (see section 2.2) and used C-SVM algorithm with a radial basis function (RBF) kernel (LIBSVM version 2.81 software package [23]). We then used a hydrophobicity plot to construct a feature vector from a protein sequence in the following two-step procedure: (1) the 20 standard amino acids were represented according to a widely applied hydrophobic scale (Kyte-Doolittle scale [24]), and (2) a fixed-length sliding window moved along the numerical vector obtained in step (1) one residue at each time and the mean value within the sliding window was calculated at each position, eventually yielding a feature vector consisting of these mean values. GPI-SVM performance was optimized by determining the optimal input length of protein sequences, the size of the sliding window, and the optimal values of the regularization parameter and the kernel parameter successively. We initially held 20 residues from the C-terminus for each protein in the training dataset as the input length and then increased the length of the input sequence one residue at a time until it reached 80 residues. Throughout the above process, window size was set to 9 residues and the default values of LIBSVM were used for C and γ . We chose as the optimal input length the one that corresponded to the highest prediction accuracy of the SVM classifier under the 5-fold cross validation (CV) test. Subsequently, by using the optimal

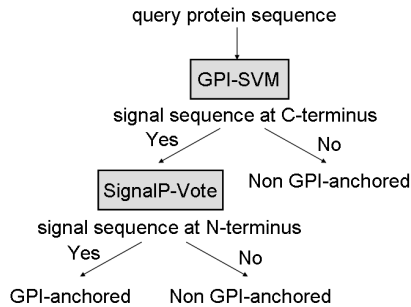


Fig. 1. Flowchart of the GPI-SVM+SignalP-Vote prediction protocol

input length and default C and γ , a series of sliding window size (odd numbers ranging from 1 to 21) were investigated with regard to prediction accuracy (also under the 5-fold CV test) and the value of area under the receiver operating characteristic curve (AUC). Given the optimal input length and window size, optimization of the parameters of the classifier is equal to trying to find the highest accuracy of the 5-fold CV test in the search space of C and γ . We used the population-based stochastic optimization technique called Particle Swarm Optimization (PSO) [25] to tune C and γ .

The SignalP-Vote is a voting system which consists of seven binary indicators obtained from SignalP version 3.0 [26]. Five of the seven indicators belong to the SignalP-NN and the other two belong to the Signal-HMM. The voting system uses a simple majority-voting strategy: a protein is considered to have the N-terminal signal only if that decision is supported by more than three voters. Two widely used measures, precision and recall, were used for SignalP-Vote to evaluate the quality of classification. Precision and recall are defined as $\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$ and $\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$, where the terms TP, TN, FP and FN mean true positives, true negatives, false positives and false negatives. To compare SignalP-Vote and single indicator, we use F-score, which is defined as $\text{F-score} = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$. F-score thus is a weighted mean of precision and recall, where it reaches best value at 1 and worst value at 0.

2.2 Datasets

We collected 587 GPI-anchored proteins from UniProKB/Swiss-Prot by selecting those labeled "GPI-ANCHOR" or "GPI-LIKE-ANCHOR" in the KW field in the year of 2005. They covered a wide range of kingdoms, and in the present study we used the 520 of them that had been annotated with definite comments about the GPI-anchored modification site (ω -site) in the field of feature table (FT) and that had a sequence length equal to or greater than 100 residues. The negative dataset provided by Pascal Mäser [15] includes 441 proteins that had been selected from GeneBank by text-based searching. The sequence similarity between any two of them was less than 50%. We selected the 429 proteins that have a sequence length equal to or greater than 100 residues. They included 105 cytosolic proteins, 63 secreted proteins, 104 N-TM-C proteins (transmembrane proteins with a hydrophobic C-terminus and an N-terminal export signal predicted by SignalP), and 157 other transmembrane proteins. The SVM training dataset thus consisted of 520 positive samples and 429 negative samples.

The performance of SignalP-Vote was compared with that of using single indicator obtained from SignalP. To this end, we used the test set of eukaryotic secretory and non-secretory proteins those were selected by Menne et al. [27] from Swiss-Prot database. This test set consisted of 1158 positive samples and 1142 negative samples and are available at <ftp://ftp.ebi.ac.uk/pub/contrib/swissprot/testsets/signal>.

To evaluate the performance of our prediction protocol on the proteomic scale, we used the proteome of *S. cerevisiae* because it has been widely studied in experiment. We retrieved the proteome of *S. cerevisiae* (6718 protein sequences) from

the Stanford Genome Database (SGD, ftp://genome-ftp.stanford.edu/pub/yeast/data_download/sequence/genomic_sequence/orf_protein), which were translated from open reading frames (ORFs).

Searching the feature table (FT) fields of *S. cerevisiae* proteins in UniProtKB /Swiss-Prot, we found 40 proteins that possess definite comments about ω -site. We noted that fifteen of 40 proteins were not included in the SVM training dataset as described. This 40-protein dataset was used as a sensitivity test for comparing our prediction protocol with others proposed for screening GPI-anchored proteins.

3 Results and Discussion

3.1 Results

When the number of input residues was increased as described in section 2.1, the accuracies of 5-fold CV test increased until the input length reached 60-64 residues (Fig. 2).

Therefore, 60 residues counted from the C-terminus were selected as the optimal input length. Table 1 shows that prediction accuracy and AUC value were highest (respectively 95.50% and 0.96) when the size of the sliding window size was 9 residues.

Table 1 also shows that the accuracy difference between window sizes 1 and 3 is greater than that between any other consecutive pairs of window sizes. The optimal parameters C and γ obtained by using the PSO technique were 23.20 and 0.051, respectively. Compared with the use of default C and γ (1.0 and 0.019), GPI-SVM raise the accuracy of 5-fold CV test by 0.5% (96%) and AUC value by 0.01. With respect to the N-terminal signal identification, SignalP-Vote measured with Menne's test dataset has *recall* of 98.88%, *precision* of 91.82%, and an *F-score* of 0.952 (Table 2). In terms of *F-score*, SignalP-Vote is better than any single indicator.

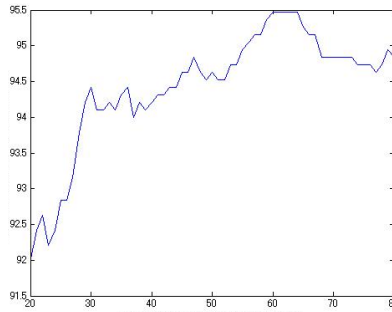


Fig. 2. The relation between accuracy and length of input sequence. The numbers on the abscissa means Length of input protein sequence and the number on the ordinate means accuracy under 5-fold CV test.

Table 1. Window size vs. Prediction accuracy*

Window size	Accuracy(%)	AUC
1	70.60	0.7663
3	94.63	0.9570
5	95.36	0.9606
7	95.47	0.9568
9	95.50	0.9631
11	94.94	0.9563
13	94.73	0.9498
15	94.63	0.9561
17	94.63	0.9568
19	94.63	0.9533
21	94.42	0.9579

* The default parameters C and γ provided by LIBSVM were used, where C was set to one and γ was the reciprocal of the length of a feature vector.

The results obtained using four prediction protocols to screen the proteins in the sensitivity test dataset are summarized in Table 3. With respect to the sensitivity test dataset, GPI-SVM+SignalP-Vote and GPI-SOM+SignalP-HMM [15] have the same sensitivity of 95%(38/40) and are better than the other two, Big-PI+SignalP-Vote (40%) and GPI-SOM+Phobius (90%). GPI-SVM+SignalP-Vote could not predict two proteins, YP056_YEAST and SPS2_YEAST, as GPI-anchored proteins. These wrong predictions were caused by the wrong classifications of GPI-SVM and SignalP-Vote, respectively. GPI-SOM+SignalP-HMM could not predict two proteins, SPS2_YEAST and MKC7_YEAST, as GPI-anchored proteins. The wrong prediction of SPS2_YEAST was due to SignalP-HMM, while the wrong prediction of MKC7_YEAST was due to GPI-SOM. And 14 of 15 newly annotated proteins (names in bold font in Table 3), which were not included in the training dataset, were identified by our protocol.

Our new protocol as well as GPI-SOM+SignalP-HMM were then used in a proteome-wide investigation on the proteome of *S. cerevisiae* (Table 4). GPI-SVM found 298 of the total 6718 protein sequences to have the C-terminal signal of GPI-anchored proteins, and SignalP-Vote found 105 of this subset to have the N-terminal signals. After manually removing protein sequences translated from the dubious open read frames (ORFs), we found GPI-SVM+SignalP-Vote to have identified 88 of 105 proteins as GPI-anchored proteins. Of these 34 experimentally identified GPI-anchored proteins [28], 20 proteins were not covered by the GPI-SVM's training dataset; 19 of these 20 proteins were identified by our protocol. There were 670 proteins possessing potential C-terminal signal were selected by GPI-SOM from the whole proteome of *S. cerevisiae*. One hundred sixty-seven of 670 proteins were then presumed by SignalP-HMM to have the N-terminal signals of GPI-anchored proteins. After the dubious ORFs sequences were removed from the 167 proteins, 124 proteins remained. Thus, only using the

Table 2. Comparison of SignalP-Vote with single indicator

Method*	Recall(%)	Precision(%)	F-score
SignalP-HMM(Cmax)	80.66	97.60	0.883
SignalP-HMM(Sprob)	91.71	93.90	0.928
SignalP-NN(Cmax)	97.24	84.34	0.903
SignalP-NN(Ymax)	98.96	91.24	0.949
SignalP-NN(Smax)	98.70	88.06	0.931
SignalP-NN(Smean)	98.19	90.38	0.941
SignalP-NN(D-score)	98.70	91.37	0.949
SignalP-Vote	98.88	91.82	0.952

* The outputs from SignalP 3.0 comprise seven values denoted as indicators here, two of which are generated by SignalP-HMM and five of which are given by SignalP-NN. A C-score, which is a measure of being part of the signal peptide, and an S-score, which indicates a "cleavage site", are reported for each position in the query sequence. Smax: the maximal S-score; Cmax: maximal C-score; Ymax: derivative of the C-score combined with the S-score; Smean: mean of the S-score; D-score: average of the Smean and Ymax score; Sprob: probability that the query sequence contains a signal peptide or not.

C-terminal signal prediction, GPI-SOM identified twice as many GPI-anchored proteins as GPI-SVM did.

Measuring the overlap of the two datasets of proteins identified by GPI-SVM+SignalP-Vote and GPI-SOM+SignalP-HMM, we found that although the two protocols had the same sensitivity, their 88-protein and 124-protein datasets shared 70 predicted proteins. The remaining 54 proteins identified by GPI-SOM+SignalP-HMM were analyzed by searching the UniProtKB/Swiss-Prot database for feature descriptions in detail. Twenty-nine of the 54 proteins had been labeled as proteins possessing transmembrane domain(s) even though 10 of the 70 proteins identified by both GPI-SOM+SignalP-HMM and our protocol were found to be annotated as transmembrane proteins. In contrast, 10 of 18 proteins identified by GPI-SVM+SignalP-Vote were found to be annotated as transmembrane proteins. Overall, GPI-SOM+SignalP-HMM had more false positives than GPI-SVM+SignalP-Vote did.

3.2 Discussion

Our new computational protocol is a powerful tool for the detection of GPI-anchored proteins. On the whole, this study shows that machine-learning techniques are more promising than the other methods for identification of GPI-anchored proteins. Compared with the previous prediction protocols, our protocol showed higher sensitivity (95%) on the sensitivity test dataset of 40 known GPI-anchored proteins selected from *S. cerevisiae*. Caro et al. predicted 32 of

Table 3. Name list of the sensitivity test dataset

Entry name	Locus name	Length	ω -site	Amino acid
CCW12_YEAST [◊]	YLR110C	133	112	G
CCW14_YEAST [◊]	YLR390W-A	238	220	N
CRH1_YEAST	YGR189C	507	483	G
CRH2_YEAST [◊]	YEL040W	467	445	N
CWP1_YEAST [◊]	YKL096W	239	217	N
CWP2_YEAST	YKL096W-A	92	71	N
DAN1_YEAST	YJR150C	298	282	N
DAN4_YEAST	YJR151C	1161	1146	N
DCW1_YEAST	YKL046C	449	425	S
DFG5_YEAST [◊]	YMR238W	458	434	D
FIT1_YEAST	YDR534C	528	512	A
FIT2_YEAST [◊]	YOR382W	153	124	S
FLO1_YEAST	YAR050W	1537	1514	G
GAS1_YEAST	YMR307W	559	528	N
GAS2_YEAST	YLR343W	555	531	D
GAS3_YEAST	YMR215W	524	506	S
GAS4_YEAST	YOL132W	471	449	A
GAS5_YEAST	YOL030W	484	452	S
HPF1_YEAST	YOL155C	967	946	A
KRE1_YEAST [◊]	YNL322C	313	288	N
MKC7_YEAST ^{+,×}	YDR144C	596	575	N
MUC1_YEAST	YIR019C	1367	1346	G
PST1_YEAST	YDR055W	444	419	N
SAG1_YEAST	YJR004C	650	627	G
SED1_YEAST [◊]	YDR077W	338	318	N
SPS2_YEAST ^{*,+,×}	YDR522C	502	475	N
TIP1_YEAST [◊]	YBR067C	210	188	G
TIR1_YEAST [◊]	YER011W	254	233	N
TIR4_YEAST [◊]	YOR009W	487	465	N
YC048_YEAST	YCL048W-A	79	55	G
YD134_YEAST ^{◊,×}	YDR134C	136	115	G
YD24B_YEAST	YDR524C-B	66	42	G
YL040_YEAST	YLR040C	224	203	N
YL042_YEAST	YLR042C	161	139	G
YL194_YEAST [◊]	YLR194C	254	232	N
YO214_YEAST [◊]	YOR214C	236	212	N
YP056_YEAST ^{◊,*,×}	YPL056C	101	81	S
YPS1_YEAST	YLR120C	569	548	N
YPS3_YEAST [◊]	YLR121C	508	483	N
YPS6_YEAST	YIR039C	537	515	N

Entry names in bold were not included in the GPI-SVM training dataset.

* Predicted incorrectly by GPI-SVM+SignalP-Vote.

× Predicted incorrectly by GPI-SOM+Phobius (Dora, <http://genomics.unibe.ch/dora/>).

+ Predicted incorrectly by GPI-SOM+SignalP-HMM.

◊ Predicted correctly by Big-PI+SignalP-Vote.

Table 4. Investigation on the proteome of *S. cerevisiae*

Prediction protocol	# of Cterm ^c	# of Nterm ^d	# of putatives ^e
GPI-SOM+SignalP-HMM ^a	670	167	124
GPI-SVM+SignalP-Vote ^b	298	105	88

Of the total 6718 protein sequences:

^a 84 proteins were ignored since their sequence lengths are all less than 32 residues and 39 proteins were not predicted by GPI-SOM.

^b 187 proteins were ignored since sequence length of each of the proteins are not satisfied the requirement of GPI-SVM (less than 60 residues).

^c Number of proteins with the C-terminal signal.

^d Number of putative GPI proteins (include dubious sequences).

^e Number of putative GPI proteins.

the 40 proteins in our sensitivity test dataset by using their NN method. De Groot et al. predicted other 32 proteins of the 40 proteins. Although the results of Caro et al. and De Groot et al. had different false positives, their protocol each has a sensitivity of 80% (32/40). The prediction protocol combining Big-PI with the SignalP-Vote had a sensitivity of 39%. Big-PI uses a scoring function for measuring amino acid composition around the GPI-anchoring site and also uses the probability of an empirical distribution based on the scores. Since a small size dataset (254 sequences) is used to estimate the parameters of the empirical distribution, it may not be suitable for our sensitivity test. GPI-SOM+SignalP-HMM adopted a machine learning technique, self-organizing map, for identifying the C-terminal signal and had 95% sensitivity that matched our protocol. When we applied GPI-SOM+SignalP-HMM and our protocol to the proteome of *S. cerevisiae*, GPI-SOM+SignalP-HMM gave more false positive predictions than our protocol did. We owe our success to the feature extraction method from protein sequences in our protocol except for the consideration of SVM's known generalization ability. We will discuss two characteristics in our feature extraction method, i.e., the window size and the length of input sequence.

The window size of more than one residue seems to be useful for improving the accuracy of GPI-anchored proteins identification. Caras and Weddell's experimental result [16] suggests that GPI attachment depends on the overall hydrophobicity of the C-terminal signal rather than specific sequences. In Fankhauser and Mäser's work, they had also tried to only use relative hydrophobic value (Kyte-Doolittle scale) of 32 positions at the C-terminus to transform the protein sequence into the numerical sequence; in other words, the window size was one residue. However, the prediction accuracy was only about 83%. In this study, we found that a window size greater than one residue can significantly improve the performance of GPI-SVM (Table II).

A longer input sequence length may reduce false positives caused by the integral membrane in spite of no direct evidence. It is just as pointed out by Fankhauser and Mäser [15] that integral membrane proteins with a trans-membrane domain at their C-termini cause many false positive predictions. To

identify the C-terminal signal, 15 residues (from a hydrophobic domain) proximate to the C-terminus of the protein are necessary and indispensable [29]. The determination of the optimal input length carried out in this study indicates that the C-terminal signal can be identified more precisely by using more residues at the C-terminus. A protein is numerically represented by selecting 22 discrete positions at the C-terminus of the protein in GPI-SOM. In addition to using the relative hydrophobic value, GPI-SOM uses other two attributes to generate the numerical representation for a protein. In contrast, 60 residues counted from the C-terminus were taken as input in GPI-SVM. The shorter input length possibly makes it difficult to distinguish GPI-anchored proteins with the integral membrane domains. We therefore assume that the input sequence length may be another reason why GPI-SOM identified about 10% proteins of the proteome as GPI-anchored proteins whereas a few proteins were identified by GPI-SVM. Experimental results [30] suggest that the length of C-terminal signal for the GPI attachment is 29-37 residues. This sequence length is less than the optimal input sequence length (60 residues). Although no experimental information is available, we assume that there would be some patterns that would have not been recognized in the region beyond the 29-37 residues at the C-terminus.

To improve the ability of the N-terminal signal prediction, we designed SignalP-Vote on the consideration that the indicators (or classifiers) may provide complementary information. Measured with Menne's test dataset, SignalP-Vote showed better performance than any single indicator did. We therefore consider it to be another feature for our new protocol.

Over last decade there have been many attempts to identify GPI-anchored proteins by using computer-based prediction protocols based on protein primary structure, and the number of identified GPI-anchored proteins stored in databases has been increasing continually. To identify GPI-anchored proteins on the proteomic scale, we have developed a computational protocol that uses an optimized classifier trained with the SVM algorithm to identify the C-terminal signal and uses a majority voting system based on the SignalP method to identify the N-terminal signal. To represent the characteristics of the overall hydrophobicity of the C-terminal signal, we selected as input 60 continuous positions counted from the C-terminus and depicted the hydrophobicity of the residues in these options by using a series of average values with a fixed-size sliding window (9 residues). Our protocol has remarkable generalization ability by virtue of support vector machine algorithm and we believe our new protocol will be a helpful tool to identify GPI-anchored proteins on the proteomic scale.

Acknowledgments

We would like to offer our sincere thanks to Professor Pascal Mäser for providing us with the negative data set for training GPI-SOM.

References

1. Powell, S.K., Cunningham, B.A., Edelman, G.M., Rodriguez-Boulan, E.: Targeting of transmembrane and GPI-anchored forms of N-CAM to opposite domains of a polarized epithelial cell. *Nature* 353, 76–77 (1991)
2. Ikezawa, H., Yamanegi, M., Taguchi, R., Miyashita, T., Ohyabu, T.: Studies on phosphatidylinositol phosphodiesterase (phospholipase C type) of *Bacillus cereus*. I. purification, properties and phosphatase-releasing activity. *Biochim. Biophys. Acta.* 450, 154–164 (1976)
3. Rijnboutt, S., Jansen, G., Posthuma, G., Hynes, J.B., Schornagel, J.H., Strous, G.J.: Endocytosis of GPI-linked membrane folate receptor- α . *J. Cell Biol.* 132, 35–47 (1996)
4. Stahl, N., Baldwin, M.A., Hecker, R., Pan, K.M., Burlingame, A.L., Prusiner, S.B.: Glycosylinositol phospholipid anchors of the scrapie and cellular prion proteins contain sialic acid. *Biochemistry* 31, 5043–5053 (1992)
5. Cashman, N.R., Loertscher, R., Nalbantoglu, J., Shaw, I., Kascsak, R.J., Bolton, D.C., Bendheim, P.E.: Cellular isoform of the scrapie agent protein participates in lymphocyte activation. *Cell* 61, 185–192 (1990)
6. Udenfriend, S., Kodukula, K.: Prediction of omega site in nascent precursor of glycosylphosphatidylinositol protein. *Methods Enzymol.* 250, 571–582 (1995)
7. Udenfriend, S., Kodukula, K.: How glycosylphosphatidylinositol-anchored membrane proteins are made. *Annu. Rev. Biochem.* 64, 563–591 (1995)
8. Caro, L.H., Tettelin, H., Vossen, J.H., Ram, A.F., van den Ende, H., Klis, F.M.: In silico identification of glycosyl-phosphatidylinositol-anchored plasma-membrane and cell wall proteins of *Saccharomyces cerevisiae*. *Yeast* 13, 1477–1489 (1997)
9. von Heijne, G.: A new method for predicting signal sequence cleavage sites. *Nucleic Acids Res.* 14, 4683–4690 (1986)
10. Nuoffer, C., Horvath, A., Riezman, H.: Analysis of the sequence requirements for glycosylphosphatidylinositol anchoring of *Saccharomyces cerevisiae* Gas1 protein. *J. Biol. Chem.* 268, 10558–10563 (1993)
11. De Groot, P.W., Hellingwerf, K.J., Klis, F.M.: Genome-wide identification of fungal GPI proteins. *Yeast* 20, 781–796 (2003)
12. Nielsen, H., Engelbrecht, J., Brunak, S., von Heijne, G.: Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein Eng.* 10, 1–6 (1997)
13. Nakai, K., Kanehisa, M.: A knowledge base for predicting protein localization sites in eukaryotic cells. *Genomics* 14, 897–911 (1992)
14. Eisenhaber, B., Schneider, G., Wildpaner, M., Eisenhaber, F.: A sensitive predictor for potential GPI lipid modification sites in fungal protein sequences and its application to genome-wide studies for *Aspergillus nidulans*, *Candida albicans*, *Neurospora crassa*, *Saccharomyces cerevisiae* and *Schizosaccharomyces pombe*. *J. Mol. Biol.* 337, 243–253 (2004)
15. Fankhauser, N., Maser, P.: Identification of GPI anchor attachment signals by a Kohonen self-organizing map. *Bioinformatics* 21, 1846–1852 (2005)
16. Caras, I.W., Weddell, G.N.: Signal Peptide for Protein Secretion Directing Glycophospholipid Membrane Anchor Attachment. *Science* 243, 1196–1198 (1989)
17. Eisenhaber, B., Bork, P., Eisenhaber, F.: Prediction of potential GPI-modification sites in proprotein sequences. *J. Mol. Biol.* 292, 741–758 (1999)
18. Vapnik, V.N.: *The Nature of Statistical Learning Theory*. Springer, New York (1995)

19. Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, Inc., Chichester (1998)
20. Zien, A., Ratsch, G., Mika, S., Scholkopf, B., Lengauer, T., Muller, K.R.: Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics* 16, 799–807 (2000)
21. Ding, C.H., Dubchak, I.: Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics* 17, 349–358 (2001)
22. Hua, S.J., Sun, Z.R.: Support vector machine approach for protein subcellular localization prediction. *Bioinformatics* 17, 721–728 (2001)
23. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. Software (2001), <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
24. Kyte, J., Doolittle, R.F.: A Simple Method for Displaying the Hydropathic Character of a Protein. *J. Mol. Biol.* 157, 105–132 (1982)
25. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 1998), Piscataway, NJ, pp. 69–73 (1998)
26. Bendtsen, J.D., Nielsen, H., von Heijne, G., Brunak, S.: Improved prediction of signal peptides: SignalP 3.0. *J. Mol. Biol.* 340, 783–795 (2004)
27. Menne, K.M., Hermjakob, H., Apweiler, R.: A comparison of signal sequence prediction methods using a test set of signal peptides. *Bioinformatics* 16, 741–742 (2000)
28. Hamada, K., Terashima, H., Arisawa, M., Yabuki, N., Kitada, K.: Amino acid residues in the omega-minus region participate in cellular localization of yeast glycosylphosphatidylinositol-attached proteins. *J. Bacteriol.* 181, 3886–3889 (1999)
29. Caras, I.W., Weddell, G.N., Williams, S.R.: Analysis of the Signal for Attachment of a Glycophospholipid Membrane Anchor. *J. Cell Biol.* 108, 1387–1396 (1989)
30. Moran, P., Caras, I.W.: A nonfunctional sequence converted to a signal for glycosylphosphatidylinositol membrane anchor attachment. *J. Cell Biol.* 115, 329–336 (1991)

Towards Large-Scale Molecular Dynamics Simulations on Graphics Processors

Joseph E. Davis, Adnan Ozsoy,
Sandeep Patel, and Michela Taufer

University of Delaware,
Newark, DE, 19716
{jedavis,ozsoy,sapatel,taufer}@udel.edu

Abstract. Atomistic molecular dynamics (MD) simulations are a vital tool in chemical research, as they are able to provide a view of chemical systems and processes that is not obtainable through experiment. However, large-scale MD simulations require access to multicore clusters or supercomputers that are not always available to all researchers. Recently, many have begun to explore the power of graphics processing units (GPUs) for various applications, such as MD. We present preliminary results of water simulations carried out on GPUs. We compare the performance gained using a GPU versus the same simulation on a single CPU or multiple CPUs. We also address the use of more accurate double precision arithmetic with the newest GPUs and its cost in performance.

Keywords: Molecular dynamics, GPU, CUDA.

1 Introduction

For years, graphics processing units (GPUs) have been used extensively in graphics intensive applications. Their development has been driven strongly by the economy, particularly the entertainment industry, in order to meet the ever increasing demand for faster, more detailed three-dimensional (3D) graphics in media such as movies and video games. These devices are designed specifically to alleviate the load of the main CPU by taking over the expensive operations required to render detailed 3D graphics. Most of these operations are inherently data parallel, and since GPUs have been developed with this parallelization in mind, they have the potential to become powerful tools for scientific computing. However, until recently GPU hardware has been restricted to operations specific to graphics processing, limiting their usefulness in other areas. Fortunately, recent efforts have led to the development of general purpose GPUs (GPGPUs) and language libraries such as NVIDIA's Compute Unified Device Architecture (CUDA) [1].

Recent efforts have explored the potential of GPUs for mathematical, scientific, and clinical computing applications. One study has reported coupling GPUs to magnetic resonance imaging (MRI) hardware for medical diagnostics [2]. In

the area of molecular modeling, GPUs have been applied to electrostatic potential calculation [3], ion placement [3], and simulations of van der Waals fluids and polymers [4]. Particularly in the areas of molecular modeling and molecular dynamics (MD), accelerating simulations by exploiting parallelism is a major concern. As with many other applications, MD has been adapted for use with massively parallel architectures such as compute clusters and supercomputers. In addition, special purpose hardware has been developed to meet these challenges, such as Protein Explorer systems with its large-scale integration (LSI) ‘MDGRAPE-3 chip’ [5] and Anton with its 12 identical application-specific integrated circuits (ASICs) designed for MD [6]. However, these architectures target very specific types of calculations, and in that respect are similar to early generations of GPUs. In addition, they are not widely available to most researchers. On the other hand, GPGPUs are cost effective, readily available in recent workstations, flexible, and easy to deploy.

MD simulations are an excellent target for GPU acceleration since most aspects of MD algorithms are easily parallelizable. Enhancing the performance of MD can allow the simulation of both larger time scales and larger length scales. Figure 1 illustrates some types of calculations that are possible at differing length and time scales. Ideally, simulations would be able to attain all-atom resolution on time scales of microseconds to milliseconds. With coarse-grained resolution, even longer time scales approaching seconds may be possible. However, the ultimate goal of any simulation is atomistic resolution of very large length scales over very long time scales, i.e., essentially continuum physics with atomic detail. The utilization of parallel architectures such as GPUs constitutes a step towards this goal.

This contribution will discuss ongoing work in our laboratory exploring the potential applications of GPUs to atomistic MD simulations. Section 2 discusses some background related to CUDA and some related work with MD

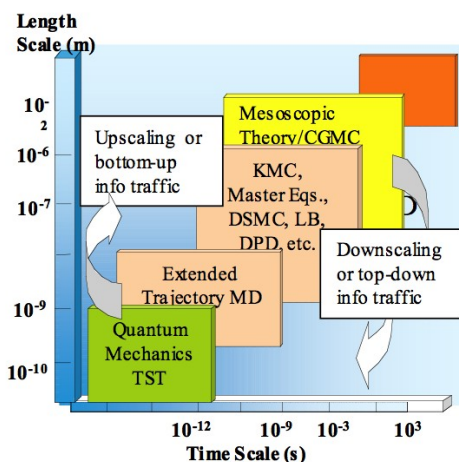


Fig. 1. An illustration of some of the types of theoretical physical calculations that are possible at varying time and length scales. Figure reproduced from Reference [7].

using CUDA. Section 3 describes in general our simulations on the CPU and the GPU. In addition, we outline our implementation of MD in CUDA. In Section 4 we provide further detail on the simulation parameters as well as present results from two case studies: pure water simulations and sodium iodide (NaI) solution simulations. Finally, in Section 5 we conclude and discuss future work.

2 Background and Related Work

2.1 Background

In the past, graphics application programming interfaces (APIs) such as OpenGL were the only means of accessing the computing resources of GPUs. In general, these APIs were not easy to use for those who were unfamiliar with graphics processing, as calculations essentially had to be “drawn” and then the resulting “images” interpreted to obtain meaningful results. Recently NVIDIA has introduced the CUDA language library. CUDA facilitates the use of GPUs by providing a minimal set of extensions to the C programming language necessary to expose the power of GPUs for general purpose applications.

GPUs are massively parallel multithreaded devices capable of executing a large amount of active threads handled by a hardware thread execution manager that overlaps computation with communication whenever possible. There are multiple streaming multiprocessors, each of which contains multiple scalar processor cores. For example, NVIDIA’s G80 GPU architecture contains 16 such multiprocessors, each of which contains 8 cores, for a total of 128 cores which can handle up to 12,288 active threads in parallel. The GPU also has several types of memory, most notably the main device memory (global memory) and on-chip shared memory accessible to all cores on a single multiprocessor. From the perspective of the CUDA programmer, the GPU is treated as a coprocessor to the main CPU. Programs are written in C and linked to the CUDA libraries. A function that executes on the GPU, called a kernel, consists of multiple threads executing code in a single instruction, multiple data (SIMD) fashion. That is, each thread in a kernel executes the same code, but on different data. Further, threads can be grouped into thread blocks. This abstraction takes advantage of the fact that threads executing on the same multiprocessor can share data via on-chip shared memory, allowing some degree of cooperation between threads in the same block.

As described above, GPU architecture is inherently different than a traditional CPU. Therefore, code optimization for the GPU involves different approaches than for the CPU. Such considerations are discussed in more detail elsewhere [1], but here we will briefly address some typical CUDA optimization strategies. First, since threads are independent, it is necessary to maximize independent parallelism, i.e., minimize the need for communication between threads. However, it is also beneficial to take advantage of the per-block shared memory, allowing a small degree of communication between threads in the same block. A balance of these two is desirable for the best performance. Second, since reading from the global memory is relatively expensive, it is often more efficient to redundantly compute a value multiple times rather than compute it once and read it from memory. In other words,

maximizing the amount of arithmetic intensive computations allows the latency of communication to be hidden by overlapping with execution. Third, when dealing with global memory, it is most efficient if reads and writes are coalesced, meaning that consecutive threads act on contiguous locations in memory. Therefore it is beneficial to structure the data and algorithms in such a way that this is possible. If this is difficult or not practical, the texture cache can be used to speed up reads from memory locations that are not contiguous. Finally, transfer between the GPU global memory and the main memory of the CPU is extremely slow compared to transfer within the GPU, so this should be done only when absolutely necessary.

2.2 Related Work

In this section we will briefly review some related work involving the implementation of MD on GPUs. One of the first instances reported in the literature was a study by Yang *et al.* [8] in which an MD algorithm was programmed for a GPU using OpenGL, as general purpose GPU programming platforms such as CUDA were not yet readily available. This required translation of the MD algorithm into graphics operations. For example, values such as the atomic coordinates were encoded in the color values of pixels. However, their application to periodic, crystalline solids made neighbor list updates unnecessary, a simplification that is not generalizable to the more fluid, less structurally ordered systems of interest to our work. Overall, this study provides a proof of concept demonstrating that implementing MD on GPUs is feasible.

Since the work of Yang *et al.* there have been several studies of MD using CUDA. Stone *et al.* [3] have integrated CUDA into NAMD, adapting nonbonded force calculations for GPU while retaining the CPU implementations of all other computations. However, minimizing the transfer of data from the GPU will likely enhance performance. This was the approach of Anderson *et al.* [4], who constructed MD code from scratch in which all computations were carried out on the GPU. Their implementation was applied to simple Lennard-Jones liquids and polymer systems with Lennard-Jones and bonded potentials only. A very similar study was carried out by van Meel *et al.* [12], with the most significant difference being the use of a cell-based list structure vs. a neighbor list structure as in Anderson's study. In addition, the study by van Meel compared CUDA with Cg, a less flexible programming language for graphics processing. Our implementation is most similar to that of Anderson *et al.* in that all aspects of the MD algorithm are carried out on the GPU using a neighbor list structure for the nonbonded calculations. Unlike Anderson, we include angle and electrostatic potentials, which are necessary to simulate solvent systems such as water.

3 Methodology

3.1 Molecular Dynamics on CPU

Several molecular modeling packages are available, with CHARMM [13], GRO-MACS [14], and NAMD [15] among the most popular. CHARMM is one of the

oldest modeling packages available, having been developed for over two decades. Today, CHARMM still has an active development community, with new simulation algorithms being incorporated constantly. As a result, CHARMM has a wide variety of algorithms, force fields, and simulation methods available compared to other packages. In addition, CHARMM has been extensively tested and validated in the many literature studies that have used it over the years. For these reasons we use CHARMM both as a model which our GPU implementation emulates and as the reference to validate the results of our algorithm.

3.2 Molecular Dynamics on GPU

Our current CUDA implementation is modeled after CHARMM in terms of the force field functional forms, neighbor list structure, and measurement units. Our water simulations use a modified version of the flexible SPC/Fw water model [9], which will be discussed in more detail below. Nonbonded interactions (Lennard-Jones and electrostatics) are calculated by a single kernel in which each thread iterates through the neighbor list for a single atom i and accumulates the interactions between i and all its neighbor list entries. The texture cache is used for reading the coordinates of the neighbor atoms since they are not contiguous in memory. Shifted force forms are used for the electrostatic and Lennard-Jones potentials so that both energies and forces go smoothly to zero at the cutoff r_{cut} . The Verlet list approach [10] is used to construct the neighbor list. Briefly, a list is constructed for each atom containing all atoms within a cutoff r_{list} , where $r_{list} > r_{cut}$. This way, the list only needs updating whenever an atom has moved more than $\frac{1}{2}(r_{list} - r_{cut})$. The list is constructed on the GPU as follows. Each thread checks the distance between an atom i and all other atoms, and adds to i 's neighbor list those atoms that are within r_{list} of i . This process is accelerated by having each block take advantage of shared memory using a previously described tiling approach [11].

Bond and angle potentials are computed using a similar list approach. For the bonds, each thread iterates through all atoms bonded to an atom i and accumulates the total bond forces. For the angles, each thread iterates through the atoms which are involved in an angle with i and calculates the appropriate interactions. Unlike the nonbonded lists, these lists are constructed once on the CPU at the beginning of the simulation, copied to the GPU, and never need to be modified.

4 Computational Experiments

4.1 Platforms

Three different NVIDIA GPUs were used for our simulations: Quadro FX 5600 (1.5GB memory, single precision), GeForce 9800 GX2 (2 GPUs per card, 512MB memory, single precision), and GTX 280 (1GB memory, double precision). For our first case study, the pure solvent systems, the GeForce 9800 GX2 was used.

For the second case, the ionic solutions, performance was compared between all GPUs, using double precision arithmetic on the GTX 280 and single precision on the others.

Our reference simulations were run on CHARMM on a Beowulf cluster consisting of compute nodes with Intel Xeon 5150 2.66 GHz (Woodcrest) CPUs. For the pure solvent simulations a single CPU was used, while for the ionic solution performance was compared with 1, 2, 4, and 8 CPUs.

4.2 Solvent Simulation

In our simulations we use a modified version of the flexible SPC/Fw water model developed by Wu *et al.* [9]. Briefly, the intramolecular potential is:

$$V_{intra} = \frac{k_b}{2} \left[(r_{OH_1} - r_{OH}^0)^2 + (r_{OH_2} - r_{OH}^0)^2 \right] + \frac{k_a}{2} (\theta_{\angle HOH} - \theta_{\angle HOH}^0)^2 \quad (1)$$

As described in Section 3.2, this potential is evaluated on the GPU using bond and angle lists. The general intermolecular potential consists of a Lennard-Jones potential modeling the van der Waals forces and a Coulomb potential for the electrostatic interactions:

$$V_{inter} = \sum_{i,j}^{pairs} \left(4\epsilon_{ij} \left[\left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}} \right)^6 \right] + \frac{q_i q_j}{r_{ij}} \right) \quad (2)$$

As mentioned in Section 3.2, this potential is shifted smoothly to zero at the cutoff and pairs separated by a distance greater than a cutoff r_{cut} are neglected. Furthermore, we have modified the Lennard-Jones parameters slightly to account for the fact that we neglect long-range electrostatics typically computed using an Ewald sum.

Starting configurations for our simulations were pre-equilibrated with CHARMM using the NVT (constant number of particles, constant volume, constant temperature) ensemble. Simulations were then run from these starting configurations using our CUDA MD code for GPUs using the NVE (constant number of particles, constant volume, constant energy) ensemble. A Verlet integrator with 1 femtosecond (fs) timestep was used both on the GPU and in CHARMM to integrate the equations of motion. Pure solvent systems consisted of cubic periodic boxes equilibrated to a density of 1.012 g/mL, the calculated equilibrium density for the SPC/Fw model [9]. Several different system sizes were used to determine the effect of system size on performance. Figure 2 shows one of the cubic water boxes used. To measure the performance, we simulated 100,000 MD steps using both CHARMM (on a single processor) and our GPU code. For the GPU simulations, only one type of GPU, the GeForce 9800 GX2, was used. Five system sizes were used, consisting of 233, 826, 1981, 3921, and 6845 water molecules. From the total execution time for each simulation, we determined the number of MD steps per second. This data is shown in Table 1. On average, our GPU code is $\sim 7x$ faster than CHARMM on a single CPU.

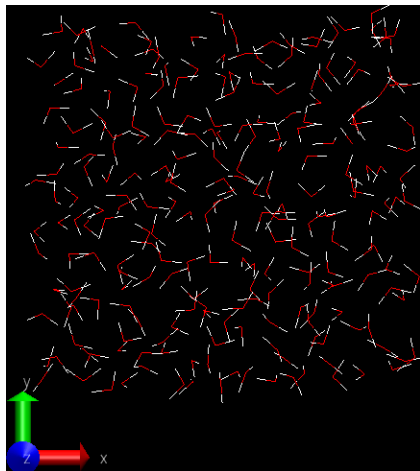


Fig. 2. A snapshot of the cubic water system containing 233 water molecules

Table 1. Performance of our GPU code for the pure water systems compared to CHARMM on a single CPU. GPU simulations were run on the GeForce 9800 GX2. The total execution time for 100,000 MD steps was used to obtain this data.

# waters	# atoms	steps/sec (CHARMM)	steps/sec (GPU)
233	699	165.34	609.09
826	2478	43.49	271.2
1981	5943	17.25	159.12
3921	11763	8.52	72.32
6845	20535	4.73	32.47

To ensure that our simulation results are accurate, we ran longer simulations (several million MD steps for a simulation time of several nanoseconds) with the 233 water system on both CHARMM and our GPU code. Figure 3 plots the temperature (a function of kinetic energy) and total potential energy over the simulation time. Both quantities fluctuate, as expected, around the same average value, indicating that our GPU implementation does in fact produce results consistent with CHARMM.

4.3 Ionic Solutions

For our second case study we simulated NaI solutions with a liquid-vapor interface. Only nonbonded interactions are applicable to the ions. Lennard-Jones parameters for sodium and iodide were transferred from the CHARMM force field and modified slightly to better reproduce *ab initio* water interaction energies [16] with our modified SPC/Fw water model. NaI solution systems consisted of a non-cubic periodic box (larger z-dimension to create a vapor interface), with simulation parameters otherwise identical to those of the pure water simulations

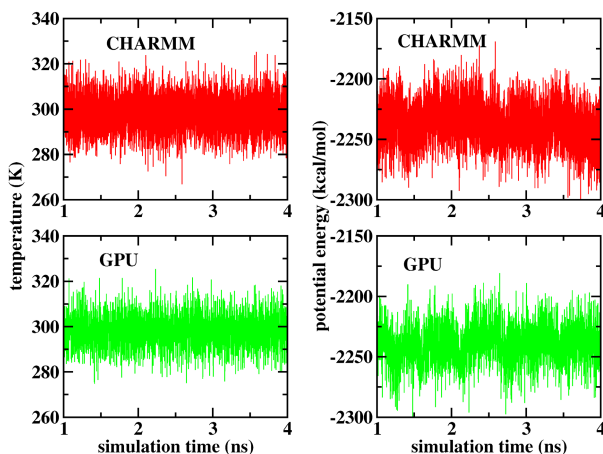


Fig. 3. Energy and temperature profiles over time for the 233 water system simulated using CHARMM and our CUDA code on a GPU

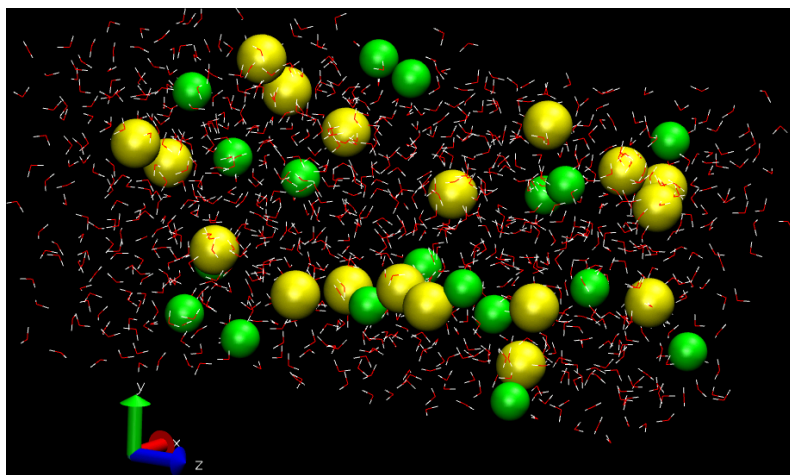


Fig. 4. Snapshot of the NaI solution system containing 988 waters, 18 Na^+ , and 18 I^-

Here we compared performance among different GPUs and different numbers of CPUs for a single system size, which consisted of 988 water molecules, 18 Na^+ ions and 18 I^- ions, for a total of 3000 atoms. Figure 4 shows a snapshot of this system. As with the previous case study we measure performance by the number of MD steps per second.

First, we compare the performance of three NVIDIA GPUs: Quadro FX 5600, GeForce 9800 GX2, and GTX 280. This data is shown in Table 2. Several observations are apparent from the data in Table 2. Both single precision GPUs perform about equally well, with the GeForce 9800 GX2 achieving $\sim 6\%$

Table 2. Performance of our GPU code for the NaI solution system compared among different GPUs and CHARMM on 1, 2, 4, and 8 CPUs. The total execution time for 10 million MD steps (10 nanoseconds simulation time) was used to obtain this data.

	steps/sec
GeForce 9800 GX2	260.88
Quadro FX 5600	246.37
GTX 280	31.79
CHARMM (1 CPU)	34.34
CHARMM (2 CPUs)	64.95
CHARMM (4 CPUs)	116.62
CHARMM (8 CPUs)	186.05

better performance. As with the pure water simulations, the speedup of the single precision GPUs over CHARMM on a single CPU is $\sim 7x$. Regarding the double precision simulation run on the GTX 280, performance drops by a factor of ~ 8 compared to the single precision. This performance is approximately equal to that of CHARMM running on a single CPU. We note, however, that not all of the optimizations we implemented with single precision were possible with double precision. Specifically, double precision does not support texture memory in the same way that single precision does. Reading coordinates from the texture memory accounts for a speedup of $\sim 2\text{--}3x$ in our single precision code. Therefore, the performance cost of double precision relative to single precision is closer to $\sim 3\text{--}4x$. Even with this consideration, the performance cost of using double precision is still significant. Finally, we note that the single precision GPUs perform better than CHARMM even on 8 CPUs. By linear extrapolation, we estimate that the performance of the single precision GPUs is approximately equivalent to that of CHARMM running on 12 CPUs.

5 Conclusions

We have presented results of all-atom MD simulations implemented on NVIDIA GPUs using CUDA. Despite the fact that our implementation is relatively naïve and straightforward, we have achieved promising results in terms of performance. Our GPU implementation performs $\sim 7x$ faster than CHARMM on a single CPU, which is a speedup approximately equivalent to CHARMM on 12 CPUs. Looking to the future, we plan to expand the MD options possible by adding potential energy terms such as dihedrals as well as additional algorithms such as Ewald summation to account for long-range electrostatics. Furthermore, we anticipate that more code optimizations are possible, leading to even more efficient performance. Finally, our ultimate goal is to integrate some GPU acceleration directly into CHARMM, which will be made possible with the upcoming FORTRAN compiler and libraries for CUDA, towards

the study of very large systems for long simulation times on the order of 100 nanoseconds.

Acknowledgments. This work was supported by the National Science Foundation, grant #SCI-0802650, DAPLDS - a Dynamically Adaptive Protein-Ligand Docking System based on multi-scale modeling, and the NVIDIA University Professor Partnership Program. The authors also wish to thank Joshua Anderson and Scott LeGrand for helpful advice.

References

1. NVIDIA CUDA Programming Guide 2.0, NVIDIA (2008)
2. Stone, S.S., Haldar, J.P., Tsao, S.C., Hwu, W.W., Liang, Z., Sutton, B.P.: Accelerating advanced MRI reconstructions on GPUs. In: Proc. of 2008 Computing Frontiers Conference, May 7-9 (2008)
3. Stone, J.E., Phillips, J.C., Freddolino, P.L., Hardy, D.J., Trabuco, L.G., Schulten, K.: Accelerating molecular modeling applications with graphics processors. *J. Comput. Chem.* 28, 2618 (2007)
4. Anderson, J.A., Lorenz, C.D., Travesset, A.: General purpose molecular dynamics simulations fully implemented on graphics processing units. *J. Comput. Phys.* 227, 5342 (2008)
5. Taiji, M., Narumi, T., Ohno, Y., Futatsugi, N., Suenaga, A., Takada, N., Konagaya, A.: Protein Explorer: A petaflops special-purpose computer system for molecular dynamics simulations. In: Proc. of 2003 ACM/IEEE Supercomputing Conference, November 15-21 (2003)
6. Shaw, D.E., Deneroff, M.M., Dror, R.O., Kuskin, J.S., Larson, R.H., Salmon, J.K., Young, C., Batson, B., Bowers, K.J., Chao, J.C., Eastwood, M.P., Gagliardo, J., Grossman, J.P., Ho, C.R., Ierardi, D.J., Kolossváry, I., Klepeis, J.L., Layman, T., McLeavey, C., Moraes, M.A., Mueller, R., Priest, E.C., Shan, Y., Spengler, J., Theobald, M., Towles, B., Wang, S.C.: Anton: A special-purpose machine for molecular dynamics simulation. In: Proc. of the 34th Annual International Symposium on Computer Architecture, June 9-13 (2007)
7. Vlachos, D.G.: A review of multiscale analysis: Examples from systems biology, materials engineering, and other fluid-surface interacting systems. *Adv. Chem. Eng.* 30, 1-61 (2005)
8. Yang, J., Wang, Y., Chen, Y.: GPU accelerated molecular dynamics simulation of thermal conductivities. *J. Comput. Phys.* 221, 799 (2006)
9. Wu, Y., Tepper, H.L., Voth, G.: Flexible simple point-charge water model with improved liquid-state properties. *J. Chem. Phys.* 124, 024503 (2006)
10. Allen, M.P., Tildesley, D.J.: *Computer Simulation of Liquids*. Clarendon Press, Oxford (1987)
11. Nguyen, H. (ed.): *GPU Gems 3*, ch. 31. Addison-Wesley, Boston (2008)
12. van Meel, J.A., Arnold, A., Frenkel, D., Zwart, S.F.P., Belleman, R.G.: Harvesting graphics power for MD simulations. *Mol. Sim.* 34, 259 (2008)
13. Brooks, B.R., Bruccoleri, R.E., Olafson, B.D., States, D.J., Swaminathan, S., Karplus, M.: CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem.* 4, 187 (1983)

14. Lindahl, E., Hess, B., van der Spoel, D.: GROMACS 3.0: A package for molecular simulation and trajectory analysis. *J. Mol. Model.* 7, 306 (2001)
15. Phillips, J.C., Braun, R., Wang, W., Gumbart, J., Tajkhorshid, E., Villa, E., Chipot, C., Skeel, R.D., Kalé, L., Schulten, K.: Scalable molecular dynamics with NAMD. *J. Comput. Chem.* 26, 1781 (2005)
16. Lamoureux, G., Roux, B.: Absolute hydration free energy scale for alkali and halide ions established from simulations with a polarizable force field. *J. Phys. Chem. B.* 110, 3308 (2006)

An Agent-Based Model of Solid Tumor Progression

Didier Dréau¹, Dimitre Stanimirov², Ted Carmichael², and Mirsad Hadzikadic²

¹ Department of Biology, University of North Carolina - Charlotte, NC

² College of Computing and Informatics, University of North Carolina - Charlotte, NC
ddreau@uncc.edu, dtstanim@uncc.edu,
tedsaid@gmail.com, mirsad@uncc.edu

Abstract. Simulation techniques used to generate complex biological models are recognized as promising research tools especially in oncology. Here, we present a computer simulation model that uses an agent-based system to mimic the development and progression of solid tumors. The model includes influences of the tumor's own features, the host immune response and level of tumor vascularization. The interactions among those complex systems were modeled using a multi-agent modeling environment provided by Netlogo. The model consists of a hierarchy of active objects including cancer cells, immune cells, and energy availability. The simulations conducted indicate the key importance of the nutrient needs of the tumor cells and of the initial responsiveness of the immune system in the tumor progression. Furthermore, the model strongly suggests that immunotherapy treatment will be efficient in individual with sustained immune responsiveness.

Keywords: self-regulation; self-organization; self-migration; agents; cell simulation.

1 Introduction

Dynamic and probabilistic modeling of tumor growth, immune responses and angiogenesis has been active areas of research in the past decade. For example, probabilistic models have been used to simulate active cell growth based on discrete cell cycle phases and durations. Continuous system models have also been used to describe cell growth in general or cell growth in terms of mathematical equations [1]. Such models vary from those that analyze the remodeling of the vasculature while ignoring changes in the tumor mass, to those that predict tumor expansion in the presence of a non-evolving vasculature [2, 3]. However, it is well accepted that vasculature remodeling and tumor growth strongly depend on one another [4, 5]. Gevertz and Torquato have developed a two-dimensional hybrid cellular automaton model of early brain tumor growth that couples the remodeling of the microvasculature with the evolution of the tumor mass [6].

However, in their modeling, tumor growth is not presented as a self-organizing complex dynamic system but rather as more random process [7]. Alternatively, the tumor mass can be recognized as a self-organized and self-regulated system. Such a system should encompass not only intrinsic properties of tumor cells (e.g., stem cells, growth, migration) but also their interactions with both the immune system and the vascular

system. Indeed, the immune system plays a major role in modulating the development of cancerous cells, with a key role in destroying tumor cells during the early phases of cancer. The blood supply providing nutrient and oxygen is also critical to cancer growth. The successful development of the tumor is tightly associated with the intrinsic tumor properties and the interactions with both the immune and the vascular systems.

In addition, to the tumor system, both the immune and the vascular system can be modeled as complex system [8-10]. For example, the self-organization of the immune system lies in immune cells' capability to produce signaling molecules (e.g., cytokines, antibodies) in response to environmental encounters, which in turn are recognized and lead to dynamic changes of their activities [11, 12]. Similarly, the blood vessel development, or angiogenesis, is controlled by multiple molecules including vascular endothelial growth factor and endothelins [13]. The secretion by tumor cells of both cytokines and angiogenic factors strongly interferes with both the local immune and angiogenic responses of the immune and vascular system [5]. The sum of the interactions between tumor, immune and vascular local systems defines both the success and the rate of tumor progression. One of the requirements for self-organization (i.e., a process of evolution where the development of new, complex structures takes place primarily in and through the system itself) is an interaction of the system with its environment. Thus, both the immune system and vasculature influence the tumor; the tumor and the vasculature influence the immune system; and tumor and the immune system influence the vasculature. Because there is no clear cause-and-effect path in the overall system, it is virtually impossible to define empirically. Thus, modeling is a plausible way to explore mechanisms by which the pieces of the system produce their effects.

Given the complexity of the interactions between tumor cells, immune cells and the vascularization during tumor progression, modeling using agent-based systems (ABS) will provide a more dynamic representation of tumor development. Therefore, here we present a new agent-based modeling of tumor progression taking into account this three-way relationship. The agent-based system approach used here uses continuous updating of agent state as the three-way interactions play out in time.

2 Materials and Methods

2.1 Software Platform

The software was implemented in NetLogo, a cross-platform multi-agent programmable modeling environment, which enables exploration of emergent phenomena, and was constructed by building on elements from Wilensky's tumor model in the NetLogo library [14]. The software permitted changing the parameters that affect tumor progression, immune system response, and vascularization, as described in the following sections. Outputs included the number of living tumor cells and the strength of the immune system.

2.2 Tumor Model

The model of De Pillis and colleagues [12, 15] was modified in the following ways. First, in the present model, tumor cells are allowed to divide, move, or die, depending

upon the presence of immune cells, and the nutrient levels associated with amount of vascularization. Second, the modifiable immune system parameter was the total number of immune system cells. By increasing the number of immune cells, the immune response is strengthened and the tumor development limited. The model allows for the control of the immune cell-cell interactions since the numbers of multiple immune cell types are parameters within this dynamic system. All those agents depend not only on their intrinsic properties but also on their environment.

2.3 Parameters

The procedure includes multiple parameters associated with tumor growth, immune response and energy availability. These parameters have been used in previous models using probabilistic, mathematical methods [11, 15]. Here, as in previous tumor models, tumor cells are allowed to divide, move, or die, depending upon energy availability (associated with vascularization) and the activity of the immune system. Specifically, the following parameters associated with tumor cells (initial number, division rate, cell death rate), immune cells (number of lymphocytes and macrophages) and the energy availability were included in an agent-based system to develop a tumor progression model.

2.4 Simulation Procedures

The simulation procedure¹ includes four components: the cell generation, the vasculature development, the proliferation routine in which individual tumor cells divide and evolve, and the immune system response. Through an iterative process, the simulation procedure is repeated at every time step or tick.

2.4.1 Cell Generation

The first step in the simulation procedure is to generate all agents that will participate in the initial run of the program. Those cells were tumor cells and two kinds of immune cells (macrophages, lymphocytes).

2.4.2 Tumor Cells

In the model, tumor cells are allowed to divide, move, or die, depending on nutrient levels, the presence of immune cells, and the passage of time. Tumor cells die in one of two ways: either they are killed by lymphocytes or macrophages, or they die due to insufficient nutrient. In this model, as in [11, 15], time is measured in simulation steps, here called “ticks” (each tick is composed of 5 equal units of time, e.g., 400 ticks = 2000 times). In the display (See Fig. 1), cells change their color depending on their age: initial tumor cells are blue small circles; very young (age less than 40 ticks) are red small circles; young (age between 40 and 70 ticks) are pink small circles; old (age between 70 and 90 ticks) are white small circles; and dead tumor cells (age above 90 ticks through apoptosis, or through immune system intervention) are grey small circles.

¹ The detailed procedure is available by contacting the authors.

2.4.3 Vasculature Development

The most common model of normal capillaries is the Krogh cylinder model, in which the capillaries are assumed to be straight, parallel vessels with uniform spacing [9]. However, images of the cerebral microvasculature [16] show that the assumption of regularly spaced, parallel capillaries is a poor approximation of the brain's capillary network. Therefore, we used a random analog of the Krogh cylinder model to generate a more physiologically relevant brain microvasculature [17]. The capillary network is allowed to exist on a triangular lattice, which is overlaid on top of the unit square containing the participating cells. In order to generate a blood vessel, a random site on the triangular lattice is chosen, as is the angle at which the vessel extends along the lattice. The vessel created is accepted as part of the vasculature and extends from its point of origin until the tissue boundary, provided that it does not violate any of the following three constraints: (1) the vessel cannot penetrate a cylinder of radius one lattice unit about an existing vessel oriented at the same angle, (2) the vessel cannot cause the intersection of three vessels at one lattice site, and (3) the vessel must vascularize at least one non-vascularized cell. Although the procedure developed take into account the development of blood supplies, in the simulated data presented here, the vascularization is simply represented (see Fig 1.) by three regions with low, moderate and high vascularization.

2.4.4 Tumor Cell Proliferation

Once the vasculature has established itself at a given time, the proliferation simulation procedure can be run. The simulation classifies the cells into one of the five types. There are two kinds of non-tumor/healthy cells: viable cells that do not actively divide and necrotic cells. There are three malignant cell types: proliferative cells that are well-vascularized and actively dividing, non-proliferative / hypoxic cells whose oxygen supply is insufficient to support cellular division and necrotic cells. The following rules are applied: (1) For each edge on the triangular lattice that holds a blood vessel, a rectangle with a length that is two times the diffusion length of oxygen (typically on the order of one millimeter) and a width equal to the length of the vessel edge is drawn. Two sides of the rectangle run parallel to the edge, and the other two run perpendicular to the edge, (2) if a cell falls within this rectangle, the cell is vascularized, and (3) if a cell does not fall within this region for any vessel in the vasculature, the cell is not well vascularized.

2.4.5 Immune System Response

The immune system is a complex system, with numerous actors and the multiple interactions involved in immune responses. The immune system exhibits a highly distributed, adaptive and self-organizing behavior with three main features: recognition, learning/ associative memory, and (specific) response [12, 18, 19]. The immune system self-regulates the populations of specific cells and their rapid increase following a challenge (in this case, tumor presence), and after eliminating the tumor cells, their decrease through the stimulation and suppression chains that it can enforce. In the present model the immune system response initial strength can be controlled. The following immune cell types are used in the current model: macrophages, and lymphocytes. The structure of immune system also varies continuously according to dynamic changes of the environment, organizing itself to adapt to the specific changes

in the surroundings [19]. In the tumor model, the cells communicate to each other through stimulation and suppression chains and work as a self and non-self recognizer. The chemical substances released from those cells (e.g., cytokines, antibodies) help them to learn from each other and to migrate to tumor site and fight tumor growth. All immune cells move randomly within the tumor microenvironment. The modeled cells are shown in Fig 1. as colored circles of two kinds: macrophages (big green), lymphocytes (yellow small). Recognition of tumor cells by lymphocytes or macrophages leads to the destruction of the tumor cell in direct contact.

2.4.6 Overall Procedure

2.4.6.1 Summary of the Procedure. Briefly, the general steps associated with the model are (1) the generation of cell population: the simulation classifies the cells as described above; (2) the generation of the vascular regions: the vascular status is arbitrarily generated by the definition of three regions with low, moderate and high vascularization. These regions are represented in Fig 1. as adjoining rectangles of different colors: low (black), moderate (blue) and high (orange) energy availability (oxygen and nutrient contents), respectively; (3) the cell proliferation: the cell proliferation is function of the cell type and surrounding environmental influences using the algorithms of De Pillis, et al. [15]; (4) Cell movements: randomly generated, resulting in contact with nutrients (in the case of cancer cells) or killing targets (in the case of immune cells); (5) Testing nutritive state: for each cell depending on its location in the vascular regions, tumor cells are allowed to divide, move, or die, depending upon nutrient levels; (6) Action of immune cells is described in previous section. Following those steps, the model status (tumor cell number and location; immune cell number and location) is updated and recorded and graphs of the number of tumor and immune cell over time is part of the output. Time is not absolute, but represents cell cycles, following the convention of De Pillis, et al. [15]. The simulations presented here are the results of 400 iterations (or 400 ticks = 2000 units of time) of the procedures described above.

2.4.6.2 Initial parameters. The tumor related parameters were introduced into the model using formula for tumor mitosis and tumor death, respectively [11, 15]. However, here the number of cells (N) in the present model was determined by the nutrient/oxygen concentrations within the cell microenvironment. The microenvironment is a factor used to calculate the energy availability (i.e., local nutrient concentration). Initially, and unless noted the following energy availability was set at 0 (low), 1.25 (medium) and 3.75 (high), respectively. The tumor cell parameters and equations were those described earlier by De Pillis et al. [11, 15]. Tumor cell division was a function of a threshold (Nutrient minimum, N_{min} set at 0.83) based on nutrient availability and on the likelihood of cell division (θ_{div} , 0.0-1.0). The number of cells dying is determined using a comparable formula with M , the number of dead cells, M_{min} (the threshold associated with a probability of cell death, set to 0.725) and the probability parameter modulating cell death (set at 0.5). The original number of tumor

cell was 2. Unless noted, the parameters associated with immune responses were initialized to macrophage (21) and lymphocytes (9).

3 Results

3.1 Modeling and Visualization of Tumor Growth

In the present model tumor progression is generated by self-organization and self-regulation of the tumor cells, the vascularization processes and the immune response behaviors. Different parameter sets were tested in the model and the effects on morphology of changing cell consumption rate, angiogenesis and immune response were measured. Four sets are presented in Fig. 1.

Tumor cells with low nutrient needs (low consumption rate $\theta_n = 0.01$, panel A) grew in density without moving far from the initial site of formation. However, tumor cells with higher need for nutrient grew more slowly (panel B, $\theta_n = 0.5$). Furthermore, to divide, the cells moved away from the initial site of the tumor, toward the regions of higher nutrient concentration, leading to two “fingered” areas of growth. Areas with low nutrients were associated with limited mitosis and subsequent death (grey tumor cells in black part of panel). The tumor evolves into a branched structure with cells actively searching for new areas with higher nutrient levels, leaving behind them necrotic debris caused by tumor cell death due to a lack of nutrients. As the intrinsic nutrient needs of the particular tumor cell increase, both the migration toward microenvironment with high nutrient content increase and the size of the tumor mass decrease (Panel C: $\theta_n = 0.75$, and Panel D: $\theta_n = 1.0$).

3.2 Cell Numbers Associated Food Consumption

A quantitative view of immune cells and tumor cells associated with the conditions presented in Fig 1 is displayed in Fig 2. Tumor cell numbers are plotted in panels A-D, immune cells in E-H, for 400 iterations (or ticks and there are 5 time units per tick).

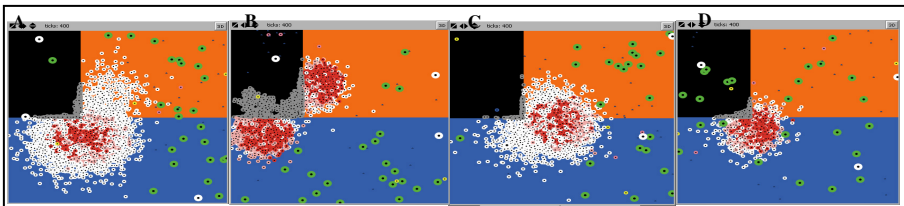


Fig. 1. Tumor mass following 400 iterations (ticks) of the ABS tumor model using the initial parameters (for more details, see the material and methods section) with 4 different nutrient consumption rates $\theta_n = 0.01$ (A), 0.5 (B), 0.75 (C) and 1.0 (D). The background represents zones of low (black), medium (blue) and high (orange) nutrient levels. Larger cells are immune cells (macrophages in green initially set at 21 and lymphocytes in yellow initially set at 9). Tumor cells are small circles of different colors depending on their age and whether they are dead; see text for explanation.

Table 1. Effects of increasing nutrient needs on the initiation of tumor growth (cell numbers above 50), simulated tumor burden and associated slope coefficient (from 1st tick with a cell number above 50 to 400 ticks). Results are presented as mean ± SE.

Nutrient needs	Cell # >50 at tick#	Tumor burden (# cells)		Slope
		200 ticks	400 ticks	
0.01	14±6.0	3901± 799	8593±1337	0.95
0.25	13±2.7	3065±1001	5446±2159	0.76
0.5	10±2.7	4059±1145	11631±1127	0.55
0.75	93±7.0	2690± 304	6470± 564	0.30
1.00	100±22	864± 255	3161± 951	0.25

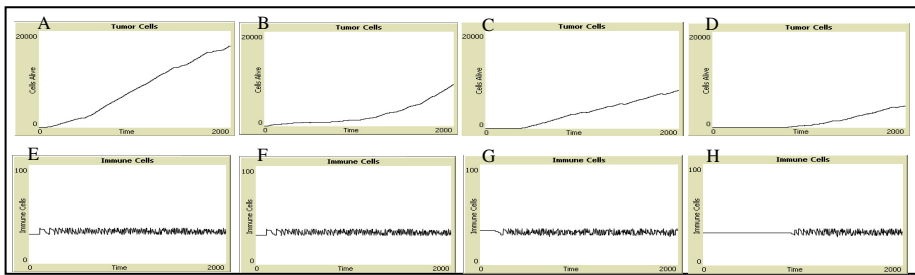


Fig. 2. Number of tumor cells (A-D) and immune cells (E-H) over time (400 ticks, there are 5 time units per tick) for experiment performed under increasing tumor nutrient needs 0.01 (A, E), 0.5 (B, F), 0.75 (C, G) and 1.0 (D, H). The conditions are those described in Fig 1.

With an immune response somewhat constant (around 30 cells, Fig 2 E-H), the generated plot for tumor mass with tumor cells with low nutrient needs (0.01, Fig 2A) contained a much larger number of tumor cells when compared to tumor cells with higher nutrient needs (0.75 and 1.0, Fig 2C-D). The simulations were repeated (n=3-5), and in those simplified conditions, the steepness of the slope of the cumulative number of tumor cells modeling tumor growth was inversely related to the tumor cell need of nutrients (slope coefficient = $0.9855 \times (\text{consumption rate})^{-1.0009}$, $r^2=0.9803$, Table 1.). In addition, with all input parameters being equal, the repeat of the simulation resulted in a broad variability of the number of ticks before the tumor mass reach 50 cells and tumor cell numbers after 200 and 400 ticks, respectively (Table 1.). Furthermore, the time to reach 50 tumor cells was significantly increased as the tumor cell intrinsic energy need increased.

3.3 Effects of the Initial Immune Cell Number on Tumor Growth

The simulation presented above (Fig 2 E-H) is a representation associated with different and constant immune responses. In the present model, the effect of the host immune responsiveness to tumor growth was modeled by simply setting the initial number of macrophages and lymphocytes to 21:9 (Fig 3E), 42:18 (Fig 3F), 63:27 (Fig 3G) and 84:36 (Fig 3H). In these conditions, the model confirms the importance of the initial immune responsiveness in slowing the tumor growth and limiting the tumor burden.

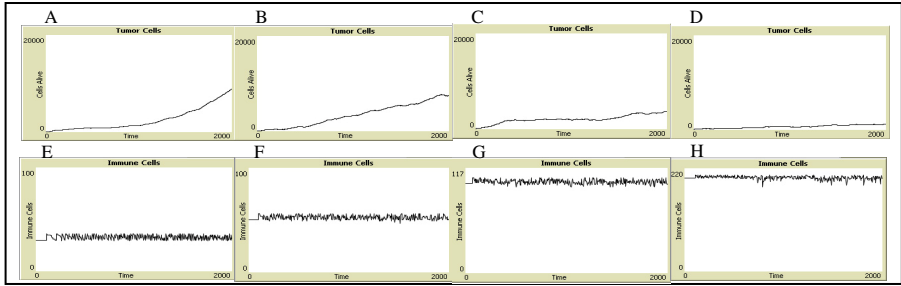


Fig. 3. Tumor burden (number of tumor cells) over time in a simulated environment with 30 (A), 50 (B), 100 (C) and 200 (D) initial immune cells, respectively.

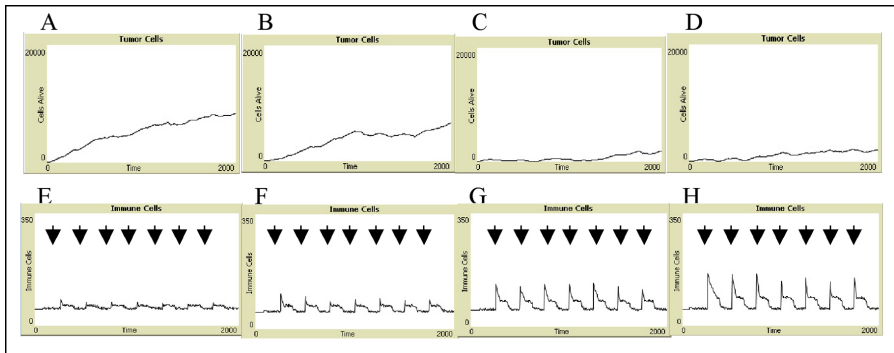


Fig. 4. Tumor burden (number of tumor cells) over time associated with increased immune responsiveness x2 (A), x3 (B), x4 (C) and x5 (D), respectively. Using the conditions described earlier (see Fig 3) the number of immune cells was modulated every 50 ticks to mimic a routine immunotherapy treatment leading to increase in immune cells within the tumor microenvironment. Arrows mark the repeated time of treatment.

The stronger the initial immune status, i.e., the number of immune cells available at the tumor location, the less tumor cells will survive independently of their environment. With 30 immune cells (Fig 3A), the tumor grows and starts to growth exponentially at about 1000 time units. With 50 immune cells present (Fig 3B), the tumor grew more or less linearly over the entire period shown here. Linear growth with lower slopes (Fig 3CD) over the time periods represented here were observed when high number of initial immune cells were used in the model.

3.4 Effects of Repeated Alterations of the Immune Cell Number on Tumor Growth

Immunotherapy is based on the correlation between host immune responses and tumor burden. Overall, the stronger the immune response, the less tumor burden as modeled earlier (Fig. 3). Therefore, we used the model to mimic the effects of a repeated immunotherapy treatment leading to increase in immune cells present within the tumor microenvironment. The modeled treatment increased the total immune cell number by 2 (Fig 4E), 3 (Fig 4F), 4 (Fig 4G) and 5 (Fig 4H) folds, respectively. As shown Fig 4., the

increase in immune cell numbers was associated with a drastic decrease in the tumor burden especially when with 4 and 5 folds the initial immune cell numbers.

4 Discussion

Computer simulations have become a useful part of investigating many natural systems, to gain insight into the operation of those systems, and to make mechanistic predictions about their behavior. Solid tumor progression in particular has been modeled extensively, but have had limited success in predicting tumor growth rate, recurrence or therapy outcome [2-5, 11, 15]. The agent-based model presented here allows the study of inputs associated with tumor intrinsic properties, the immune responsiveness and the vascularization (energy availability) in solid tumor progression. In contrast with other dynamical system approaches, the ABS approach implemented here uses time-dependent interactions. The model presented mimicked the growth of tumor mass based on the energy need of a given tumor cell clone, immune responsiveness. Furthermore, the exogenous addition of immune cells akin to an immunotherapy treatment was associated with a significant decrease in tumor burden during time the simulation was run.

The ABS model used here not only illustrates the tumor, the immune system and vascularization processes but also may provide new sets of principles on the quantitative and dynamic relationships among those systems. Each component within such a system carries out a simple task, but as a whole such systems are able to carry out much more complex tasks than would be possible by the components acting alone. Such behavior emerges in a coherent way through the local interactions of the various components. These systems are particularly robust, because they adapt to the environmental changes, and are able to ensure their own maintenance or repair. By knowing more about the system and the properties of the individual agents the tumor model can be extended and will be able to mimic individual tumor growth closely. The reliability and the trust people put in computer simulations depends on the validity of the simulation model. Although the simulations presented appear to mimic the development of a generic solid tumor, the model is based on assumptions that overall simplify the biology inherent to tumor progression. Nevertheless, the use of tumor cell intrinsic needs and immune cells within a simplified vascularization configuration still lead to realistic simulations. Furthermore, in the ABS modeling presented here due to its stochastic nature, multiple runs using the same initial parameters lead to variations in the tumor burden compatible with biological variability. Similar observations were made for the number of time unit needed to reach 50 tumor cells. However, these variations were not a fundamentally different answer for each execution even when the system is adapting to environment and behavior of individual elements. These observations indicate that within the present model, the results are reproducible. Therefore, the timing to tumor growth and the speed of the tumor burden increase may have a specific pattern that can be defined.

In the present simulation, the environment was arbitrarily divided into three different nutrient concentration levels, which is an obvious oversimplification of the “real” tumor vascularization dynamic. Nevertheless, this approach provides the bases for the development of more heterogeneous vascular environment closer to those observed in

solid tumors. Fig. 1 indicates that even with this simplification, a reasonable result was obtained, thus validating the simplification. As in previous experiments [11], the increased needs of nutrient by the tumor cells effectively lead tumor mass spreading within environment providing energy (well vascularized). Indeed, as mimicked by the present model, the vascularization of a nascent tumor mass is critical to its growth [20]. Improvement to the model will include the generation of a more realistic vascular network based on both the intrinsic properties of the tumor cells and anatomical location.

Tumor progression is also function of the immune responsiveness of the patient. Here, the model uses focuses on macrophages and lymphocytes and their effects on tumor growth. The model suggests indicates that as the number immune cells present with the vicinity of the tumor early on increase, the tumor burden increase more slowly. Although the model utilizes only macrophages and lymphocytes without distinguishing the different subsets and roles of each of the immune cell types, the simulations appears to mimic observations made in both patients and animal models [21, 22]. Furthermore, variations in the immune responses over time and between patients are the basis for cancer treatment aimed at boosting the patient immune responses [21, 22]. Such approaches have been met with only limited success in part because of a lack of full understanding of the individual and local immune tumor cells interactions [22]. The model presented here allowed simulations of alterations in number of immune cells and led to a much slower increase in the tumor burden (Fig 4.). However, similar simulations conducted using a lower initial number of immune cells led to an increase in the tumor burden (data not shown). These simulations mimic both clinical and animal studies and support the key role of the initial immune responsiveness in the development of valid immunotherapy treatments [21, 22].

Overall, the agent-based system presented, despite some oversimplifications of the system components involved (tumor, immune, vascular), provided a realistic representation of the progression over time of the tumor burden that appeared to be comparable with both animal models of solid tumors and human observations [22-24]. Indeed, the model simulated the role of the intrinsic nutrient needs of the tumor: the less nutrient a tumor cell needs, the faster it grows, and the stronger the initial immune response, the slower the tumor burden increases. Furthermore, the model also predicted the tumor burden outcome following immunotherapy treatment in patients with low and high in initial immune responsiveness, failure and success in limiting tumor growth, respectively. Additional experiments are ongoing to more fully mimic the immune microenvironment taking into account more fully the interactions between each immune cell types. Whether various alterations in both the nutrient and oxygen zone, or in the details of the immune responses will provide additional nuances to the modeling procedure is currently being investigated.

5 Conclusions

The procedure implemented allows the input of data collected from primary tumor mass as initial values for both vascularization and for some of the immune cells present within the primary tumor. Whether with additional details or as is the proposed procedure allows a decision regarding the likelihood of tumor re-growth or of

metastases remains to be assessed. Nevertheless, the procedure, as presented, takes advantage of the use of agent-based systems and strongly suggests that it may not only provide information as a diagnostic and prognostic tool but also it may clarify the multiple interactions associated with the combination of the tumor, immune and vascular system all involved in the development of solid tumor growth. Finally, repeated simulations provided consistent results with some variations in the number of tumor cells without being comparable to the extreme variation observed in cancer patients regarding tumor growth. This flexibility intrinsic to the procedure used will provide the framework to further detail the key events associated with the development of solid tumors. The variations observed in the simulations repeated under similar conditions points to the flexibility of the agent based approach in developing a tumor modeling procedure that would more closely mimic tumor growth *in vivo*. In future work more detailed immune cell populations and stress markers could be included in the model as additional factors to the presented solid tumor progression model.

Acknowledgments

The authors would like to thank David P. Bashor (UNC Charlotte) for helpful discussions and critical reading of the manuscript.

References

1. Komarova, N.L.: Mathematical modeling of tumorigenesis: mission possible. *Curr. Opin. Oncol.* 17, 39–43 (2005)
2. Grizzi, F., Russo, C., Colombo, P., Franceschini, B., Frezza, E.E., Cobos, E., Chiriva-Internati, M.: Quantitative evaluation and modeling of two-dimensional neovascular network complexity: the surface fractal dimension. *BMC Cancer* 5, 1–14 (2005)
3. Kohandel, M., Kardar, M., Milosevic, M., Sivaloganathan, S.: Dynamics of tumor growth and combination of anti-angiogenic and cytotoxic therapies. *Phys. Med. Biol.* 52, 3665–3677 (2007)
4. Geromichalos, G.D.: Importance of molecular computer modeling in anticancer drug development. *J. B. U. On.* 12(suppl. 1), S101–S118 (2007)
5. Chaplain, M.A., McDougall, S.R., Anderson, A.R.: Mathematical modeling of tumor-induced angiogenesis. *Annu. Rev. Biomed. Eng.* 8, 233–257 (2006)
6. Gevertz, J.L., Torquato, S.: Modeling the effects of vasculature evolution on early brain tumor growth. *J. Theor. Biol.* 243, 517–531 (2006)
7. Iafolla, M.A., McMillen, D.R.: Extracting biochemical parameters for cellular modeling: A mean-field approach. *J. Phys. Chem. B* 110, 22019–22028 (2006)
8. Delsanto, P.P., Condat, C.A., Pugno, N., Gliozzi, A.S., Griffa, M.: A multilevel approach to cancer growth modeling. *J. Theor. Biol.* 250, 16–24 (2008)
9. Zhang, L., Athale, C.A., Deisboeck, T.S.: Development of a three-dimensional multiscale agent-based tumor model: simulating gene-protein interaction profiles, cell phenotypes and multicellular patterns in brain cancer. *J. Theor. Biol.* 244, 96–107 (2007)
10. Mantovani, A., Marchesi, F., Porta, C., Sica, A., Allavena, P.: Inflammation and cancer: breast cancer as a prototype. *Breast* 16(suppl. 2), S27–S33 (2007)

11. de Pillis, L.G., Gu, W., Radunskaya, A.E.: Mixed immunotherapy and chemotherapy of tumors: modeling, applications and biological interpretations. *J. Theor. Biol.* 238, 841–862 (2006)
12. Mallet, D.G., De Pillis, L.G.: A cellular automata model of tumor-immune system interactions. *J. Theor. Biol.* 239, 334–350 (2006)
13. Knowles, J., Loizidou, M., Taylor, I.: Endothelin-1 and angiogenesis in cancer. *Curr. Vasc. Pharmacol.* 3, 309–314 (2005)
14. Wilensky, U.: NetLogo, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (1999), <http://ccl.northwestern.edu/netlogo>
15. De Pillis, L.G., Mallet, D.G., Radunskaya, A.E.: Spatial tumor-immune modeling. *Computational and Mathematical Methods in medicine* 7, 159–176 (2006)
16. Chen, K.C., Kim, J., Li, X., Lee, B.: Modeling recombinant immunotoxin efficacies in solid tumors. *Ann. Biomed. Eng.* 36, 486–512 (2008)
17. McGuire, B.J., Secomb, T.W.: A theoretical model for oxygen transport in skeletal muscle under conditions of high oxygen demand. *J. Appl. Physiol.* 91, 2255–2265 (2001)
18. Petruccio, C.A., Kim-Schulze, S., Kaufman, H.L.: The tumour microenvironment and implications for cancer immunotherapy. *Expert Opin. Biol. Ther.* 6, 671–684 (2006)
19. DeNardo, D.G., Johansson, M., Coussens, L.M.: Immune cells as mediators of solid tumor metastasis. *Cancer Metastasis Rev.* 27, 11–18 (2008)
20. Hanahan, D., Weinberg, R.A.: The hallmarks of cancer. *Cell* 100, 57–70 (2000)
21. Neeson, P., Paterson, Y.: Effects of the tumor microenvironment on the efficacy of tumor immunotherapy. *Immunol. Invest.* 35, 359–394 (2006)
22. Grande, C., Firvida, J.L., Navas, V., Casal, J.: Interleukin-2 for the treatment of solid tumors other than melanoma and renal cell carcinoma. *Anticancer Drugs* 17, 1–12 (2006)
23. Mukherjee, P., Ginardi, A.R., Madsen, C.S., Sterner, C.J., Adriance, M.C., Tevethia, M.J., Gendler, S.J.: Mice with spontaneous pancreatic cancer naturally develop MUC-1-specific CTLs that eradicate tumors when adoptively transferred. *J. Immunol.* 165, 3451–3460 (2000)
24. Dewan, M.Z., Terunuma, H., Takada, M., Tanaka, Y., Abe, H., Sata, T., Toi, M., Yamamoto, N.: Role of natural killer cells in hormone-independent rapid tumor formation and spontaneous metastasis of breast cancer cells in vivo. *Breast Cancer Res. Treat.* 104, 267–275 (2007)

Deciphering Drug Action and Escape Pathways: An Example on Nasopharyngeal Carcinoma

Difeng Dong¹, Chun-Ying Cui², Benjamin Mow³, and Limsoon Wong¹

¹National University of Singapore, Singapore
{dong.difeng,wongls}@comp.nus.edu.sg

²Capital Medical University, China
ccy@ccmu.edu.cn

³West Clinic Excellence Cancer Center, Singapore
bmow@westexcellence.com

Abstract. We have designed a drug pathway identification system, which we called Drug Pathway Decipherer, to generate hypotheses on drug treatment responsive pathway. Decipherer takes in both pre- and post-treatment gene expression data, and evaluates known biological pathways against the data. We applied Decipherer to two gene expression datasets of human nasopharyngeal carcinoma treated with CYC202. Results show that the identified RAS-ERK pathway and PI3K-NF κ B-IAP pathway are closely associated with treatment outcome. Decipherer is implemented in Java, and it is available together with supplementary material at <http://www.comp.nus.edu.sg/~wongls/projects/drug-pathway>.

Keywords: gene expression, drug pathway, NPC, CYC202.

1 Introduction

Biological pathways have been incorporated into gene expression analysis to understand drug treatment response in disease populations [17]. Some works focus on enrichment analysis of gene groups extracted from pathway [25, 218]. Zeeberg *et al.* [25] and Doniger *et al.* [2] use a hypergeometric test to determine statistically over-represented pathways in a list of differentially expressed genes in treatment. Subramanian *et al.* [18] propose the gene set enrichment analysis (GSEA), which uses a weighted Kolmogorov-Smirnov statistics to compare two sets of distributions and uses resampling to estimate false discovery rates (FDR). Other research groups identify responsive genetic networks under drug treatment [26, 64]. Zien *et al.* [26] exhaustively enumerate all possible gene combinations on a metabolic pathway, and select the most co-expressed gene group as the responsive pathway. Ideker *et al.* [6] follow their work by extending metabolic pathway to a protein-protein interaction network, and use an annealed random method to generate candidate gene subnetworks for statistical evaluation. Guo *et al.* [4] follow Ideker *et al.* [6], but their evaluation bases on the co-expression between interacted genes rather than the significance of expression change of genes in the identified subnetworks. A more recent work is called

PathwayExpress [3], which is a web-based application to evaluate the change of gene expression data in framework of KEGG pathway database [7]. However, most existing works fall short on several issues [17]: these works provide little information on the interplay between selected genes; the collection of pathways that can be used, evaluated and ranked against the observed expression data is limited; and the generated hypotheses are still too general to guide further research and treatment. In this paper, we present a drug pathway identification system, which we called Drug Pathway Decipherer (Decipherer), to generate hypotheses on drug responsive pathway. CYC202 (Cyclacel Ltd, Dundee, United Kingdom; Seliciclib; R-roscovitine), a CDK inhibitor, is recently studied for its anti-tumor effect in human nasopharyngeal carcinoma (NPC) cells *in vitro* and *in vivo*. 3 NPC cell lines and 13 NPC patients were treated with CYC202, and the expression of selected genes were measured during the process of treatment. Both cell lines and patients in the study responded to drug treatment differently. Our target is to understand the drug action of CYC202 in these NPC samples as well as to identify escape pathways for drug-resistant individuals. We applied Decipherer to both NPC datasets. Results reveal that RAS-ERK cell proliferation pathway and PI3K-NF κ B-IAP anti-apoptosis pathway have strong correlations with treatment outcome. Related medical assays and public publications also support our conclusions.

2 System and Methods

2.1 Overview

Decipherer is a framework for statistical evaluation of known biological pathways against gene expression data. It consists of 4 partitions distributed on two biological levels. Figure 1 shows the diagram of its workflow.

2.2 Data Source

Both NPC gene expression datasets contain 380 genes selected for apoptosis, cell proliferation, and cell cycle regulation. For the *in vitro* dataset, 3 cell lines, CNE1, CNE2 and HK1 were measured for their gene expression before the treatment of CYC202, and 2, 4, 6, 12 and 24 hours after the treatment, respectively. As a result, CNE1 responded poorly; CNE2 responded in a limited way; HK1 fully responded. For the *in vivo* dataset, 12 NPC samples and 1 non-tumor sample were taken from NPC patients. Gene expression were measured before and after the treatment of CYC202. 7 patients were reported to have molecular response to the treatment. (See supplementary material for more details of patients and cell lines.)

For pathway, we have collected 108 signaling pathways from KEGG pathway database (September 14, 2008) [7] and 49 signaling pathways from Ingenuity Pathway database (July 12, 2008) [27]. (See supplementary material for more details of the collected pathway information.)

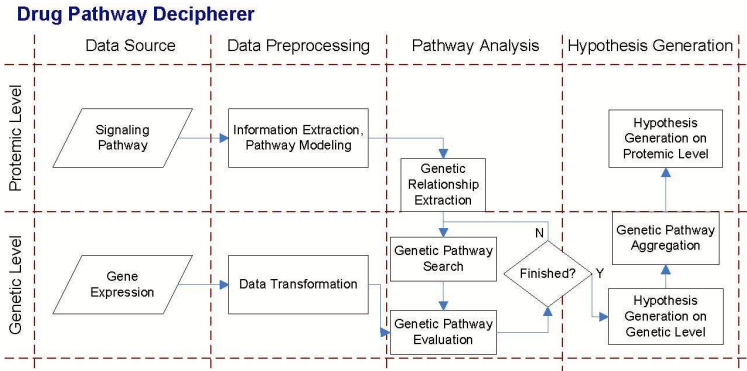


Fig. 1. The workflow of Decipherer. (a) Data source: structured signaling pathways and drug treatment gene expression data are taken as the input. (b) Data preprocessing: signaling pathways are modeled as graphs, and relative expression indicating gene expression change under drug treatment are computed. (c) Pathway analysis: pairwise genetic relationships are extracted from the modeled signaling pathways and evaluated against the relative expression data. (d) Hypothesis generation: Co-expressed genetic relationships are selected to be connected into complete genetic pathways, and statistical tools are performed to generate drug pathway hypotheses. Signaling pathway status are estimated based on the hypothesized genetic pathways.

2.3 Preprocessing Data Source

To capture gene expression change in response to drug treatment, original gene expression data are transformed into the relative expression (RE) values. Given pre- and post-treatment gene expression value e and e' , if $e' \geq e$, then RE is $e'/e - 1$; otherwise, RE is $1 - e/e'$. Intuitively, rather than log-ratio transformation, RE describes expression change in multiples in a linear scale, which allows pairwise drug effect on gene expression data to be measured by a linear correlation metric.

Signaling pathways are modeled by directed graphs. Formally, A signaling pathway γ is a directed graph (P, I) , with P the vertex set, representing the collection of proteins in pathway, and I the edge set, representing the collection of interactions between proteins. An interaction is a triplet $i = \langle p_1, p_2, s \rangle$, with $p_1, p_2 \in P$ and $s \in S$, where $S = \{\$activation, \$inhibition\}$ is the set of terms used to denote interaction type¹.

2.4 Extracting Genetic Relationships

Since protein activity can not be directly observed with gene expression data, we need to associate an interaction with one or more genetic relationships. Formally, a genetic relationship (or simply a relationship) is a triplet $q = \langle g_1, g_2, s \rangle$,

¹ Decipherer only considers the simplest interaction types. According to KEGG pathway database, we reduce other interaction types as: expression \rightarrow activation, repression \rightarrow inhibition, binding (association) \rightarrow activation, dissociation \rightarrow inhibition, ubiquitination \rightarrow inhibition.

with $g_1, g_2 \in G$ and $s \in S$, where G is the whole gene collection and S is defined in Section 2.3. To extract relationships, proteins in an interaction are mapped into their encoding genes, and the interaction type is reserved in extracted relationships. As one protein can be encoded by more than one gene, multiple relationships may be extracted from one interaction. For example, $\langle \text{Ras}, \text{Raf}, \$activation \rangle$ is an interaction. Since Ras can be encoded by either HRAS or KRAS, and Raf is encoded only by RAF1, the extracted relationships are $\langle \text{HRAS}, \text{RAF1}, \$activation \rangle$ and $\langle \text{KRAS}, \text{RAF1}, \$activation \rangle$.

2.5 Scoring a Genetic Pathway

A genetic pathway is a chain of consecutive relationships, with each of the first gene of a relationship (except for the first relationship), equaling to the second gene of its previous relationship. For example, $\langle \text{HRAS}, \text{RAF1}, \$activation \rangle$, $\langle \text{RAF1}, \text{MAP2K1}, \$activation \rangle$, and $\langle \text{MAP2K1}, \text{MAPK1}, \$activation \rangle$ construct a genetic pathway. Particularly, the first gene of the first relationship, HRAS in the example, is called the source gene; the second gene of the last relationship, MAPK1 in the example, is called the sink gene. Theoretically, a drug can target any of the known molecules in a biological system, and drug action can be terminated in any way, like with a phenotype, a mutation, or a negative feedback, which means we should not impose any requirement on the source and sink gene. However, this makes the number of genetic pathway candidates increases exponentially as the length of pathway increases. Thus, we require a source gene to be a membrane gene and a sink gene to be either a transcription factor, a feedback, or a phenotype regulator (encoding gene).

To score a genetic pathway ϑ , we first introduce the score function of relationship. Given a relationship $q = \langle g_1, g_2, s \rangle$, if gene expression are measured at multiple time points (as our *in vitro* dataset), the correlation of q is:

$$\text{Corr}(q) = \text{Corr}(\vec{r}_{g_1}, \vec{r}_{g_2}), \quad (1)$$

where $\text{Corr}(\vec{r}_{g_1}, \vec{r}_{g_2})$ is Pearson's correlation coefficient between RE vectors of g_1 and g_2 . If gene expression are only measured at two time points (as our *in vivo* dataset), then the correlation is estimated by simply comparing the post-treatment RE of the two genes:

$$\text{Corr}(q) = \text{sign}(r_{g_1}^{post} \times r_{g_2}^{post}) \times \frac{\min_{i=1,2} |r_{g_i}^{post}|}{\max_{j=1,2} |r_{g_j}^{post}|}. \quad (2)$$

$\text{Corr}(q)$ is then transformed into a z -score, $z(q)$, against sample background. $z(q)$ are then summed up over all k relationships in ϑ into an aggregated z -score, $z(\vartheta)$, for the entire genetic pathway [6]:

$$z(\vartheta) = \frac{1}{\sqrt{k}} \sum_{q \in \vartheta} (-1)^\alpha z(q), \quad (3)$$

where $\alpha = 0$ if $q.type = \$activation$; $\alpha = 1$ if $q.type = \$inhibition$, which means that if type is $\$inhibition$, genes are expected to be regulated on opposite directions.

Genes in ϑ are permuted for 10000 times to estimate the p-value of $z(\vartheta)$, denoted by $score(\vartheta)$. Intuitively, pathway score represents the consistency between a genetic pathway and expression change of genes in it.

2.6 Generating Hypotheses

Genetic pathways satisfying statistical requirement of p-value and FDR [5] are selected as genetic hypotheses. Genetic hypotheses of same signaling pathway are then integrated to generate hypotheses of signaling pathway status. For a genetic pathway ϑ , each gene g in ϑ has an impact to the sink gene, denoted by $impact_{\vartheta}(g)$. If g is an inhibitor of the sink gene, then $impact_{\vartheta}(g) = -1$; otherwise, $impact_{\vartheta}(g) = 1$. In biological system, some genes play dual functions, namely to be both activator and inhibitor to same downstream gene. However, this does not conflict with the definition of impact, since in a single genetic pathway, the role of each gene is fixed.

Thus, for a signaling pathway γ , let $\vartheta \sim \gamma$ denote the hypothesized genetic pathway ϑ in γ , and G_{ϑ} denote the gene set in ϑ . The hypothesis of signaling pathway status Z_i^{γ} at time point i is a weighted average of RE of genes in the hypothesized genetic pathways of γ , which is in formula:

$$Z_i^{\gamma} = \frac{\sum_{\vartheta \sim \gamma} \sum_{g \in G_{\vartheta}} \left(\frac{1}{|G_{\vartheta}|} \times impact_{\vartheta}(g) \times r_g^i \right)}{|\vartheta \sim \gamma|}. \quad (4)$$

3 Results and Discussion

3.1 Signaling Pathway Database

The current signaling pathway database of Decipherer consists of three parts: 49 pathways manually constructed from Ingenuity Pathway database, 24 pathways manually constructed from KEGG pathway database, and another 84 pathways automatically constructed by invoking KEGG pathway database API. These pathways include 748 distinct genes and 181949 genetic pathways can be compiled as candidates for hypothesis generation.

3.2 Application to the *in Vitro* NPC Dataset

We applied Decipherer to the *in vitro* NPC datasets with $p < 0.05$ and $FDR < 0.25$. Results are shown in Table 1. RAS-ERK cell proliferation pathway and PI3K-NF κ B-IAP anti-apoptosis pathway are observed in all 3 cell lines, which suggests CYC202 may directly access to control cell growth and cell death. Figure 2 further compares the regulation of these two pathways among the three cell lines. From the figure, ERK pathway is significantly suppressed in the responder, HK1,

Table 1. Hypothesized drug pathways by Decipherer and by PathwayExpress in three NPC cell lines

Decipherer				PathwayExpress	
Source	Pathway	p-value	FDR	Source	p-value
CNE1					
ErbB signaling pathway	BTC→ErbB4→...→ERK→Elk	5.00E-4	0.035	Leukocyte transendothelial migration	3.65E-13
Apoptosis	NGF→PI3K→...→NFkB→IAP	1.00E-3	0.035	MAPK signaling pathway	1.01E-12
Regulation of actin cytoskeleton	GF→RTK→...→PI3K→P14P5K	4.9E-3	0.11	Toll-like receptor signaling pathway	2.26E-13
MAPK signaling pathway	NF1→Ras→...→ERK→RSK2	6.00E-3	0.11	Regulation of actin cytoskeleton	3.40E-12
Focal adhesion	GF→RTK→...→Rac→Actin	9.70E-3	0.12	ErbB signaling pathway	1.10E-14
CNE2					
MAPK signaling pathway	NF1→Ras→...→ERK→Elk1	8.00E-4	0.050	Regulation of actin cytoskeleton	3.40E-12
ErbB signaling pathway	EGF→ErbB1→...→JNK→Elk	2.00E-3	0.063	Toll-like receptor signaling pathway	2.26E-13
Focal adhesion	GF→RTK→...→Rac→Actin	5.50E-3	0.12	Pathogenic Escherichia coli infection	8.69E-8
Apoptosis	TNFA→TNFR1→...→NFkB→IAP	8.60E-3	0.14	Type II diabetes mellitus	7.89E-13
p53 signaling pathway	ATM→ATR→...→CASP8→CASP3	1.49E-2	0.19	ErbB signaling pathway	1.10E-14
HK1					
GnRH signaling pathway	GnRH→GnRHR→...→MKK3/6→p38	9.00E-4	0.046	Small cell lung cancer	1.10E-14
Apoptosis	IL-1→IL-1R→...→NFkB→IAP	1.30E-3	0.046	ErbB signaling pathway	1.10E-14
MAPK signaling pathway	NGF→TrkA/B→...→ERK→Tau	2.60E-3	0.062	Gap junction	6.8E-14
Fc epsilon RI signaling pathway	Lyn→Syk→PKC	6.90E-3	0.098	Melanogenesis	1.94E-13
Regulation of actin cytoskeleton	GF→RTK→...→Rac→PAK	6.90E-3	0.098	Toll-like receptor signaling pathway	2.26E-13

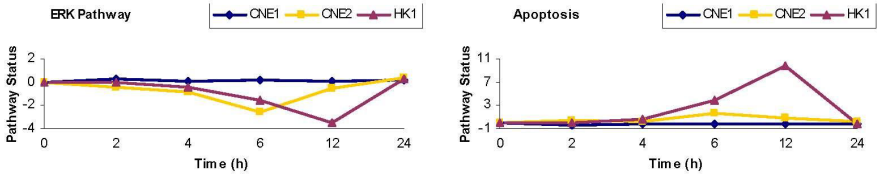


Fig. 2. Comparable status of ERK pathway and Apoptosis pathway of the 3 NPC cell lines along the treatment of CYC202.

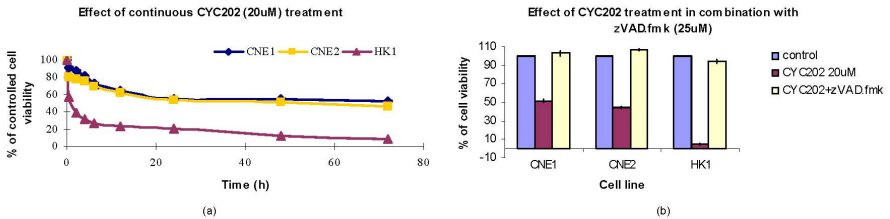


Fig. 3. Results of the associated medical assays to measure the cell viability and apoptosis level under the treatment of CYC202 for NPC cell lines: (a) The results of trypan blue test for measuring the cell viability along the drug treatment. (b) The extent of caspase-dependent apoptosis. zVAD.fmk is a caspase activity inhibitor.

but less down regulated or almost unchanged in the half-responder, CNE2, and the resister, CNE1. The apoptosis pathway is on the opposite. ERK pathway regulates cell survival, proliferation and differentiation. Suppression of this pathway represses cell viability. Therefore, the observation of ERK pathway in Figure 2 is consistent with known treatment outcome. Apoptosis pathway, on the other hand, regulates cell death. Since this pathway is induced in HK1, HK1 should be more sensitive to the treatment. To biologically prove the hypothesis of these two pathways, trypan blue test and tunel assay were used to measure the level of cell proliferation and apoptosis in these three cell lines. Results support our conclusions (Figure 3).

We compare our results with that of PathwayExpress 3 (shown in Table 1). PathwayExpress is a web-based application which evaluates gene expression data in framework of KEGG pathway database. The main issue of PathwayExpress is that it does not differentiate regulations within a signaling pathway. In Table 1, we list the top five pathways identified by each method. Results of PathwayExpress show a very high statistical significance level. However, we suspect the correctness of the p-value measurement, since 61 out of 83 signaling pathways are more significant than 10^{-4} when we applied PathwayExpress to CNE1. (see supplementary material for whole results)

3.3 Application to the *in Vivo* NPC Dataset

We applied Decipherer to the *in vivo* NPC datasets with $p \leq 0.05$ and $FDR \leq 0.25$. Table 2 gives a summary of the hypothesized post-treatment signaling pathway status (see supplementary material for genetic pathway hypotheses). Pt18 is a non-tumor sample. Other patients are classified into two groups according to their treatment response. In Table 2, since Pt18 is a normal sample and pathway status of Pt18 does not change much after treatment, we consider the pathway status of Pt18 to be a benchmark to evaluate drug response of other patients. An interesting observation is that post-treatment status of ERK pathway and apoptosis pathway in two responding groups can be almost perfectly separated by that of Pt18 (except for Pt14 in apoptosis pathway). This observation suggests suppression of ERK pathway and induction of apoptosis pathway have correlation with effective CYC202 treatment *in vivo*, and this argument also agrees with the conclusions from the *in vitro* dataset.

The leading edge analysis of GSEA selects a subset of genes that mostly differentiate two phenotype groups. Decipherer generates hypotheses of genetic pathway, which can be regarded as selecting a subset of genes from the whole gene set defined by signaling pathway. Thus, we compare Decipherer with the leading edge analysis of GSEA. The same RE values were taken as input to GSEA. All parameters were remained with default values. Gene sets were extracted from Decipherer pathway database.

As shown in Figure 4, we report the results of comparison on apoptosis pathway since it is one of our main concerned pathways in this study, yet it is identified by GSEA in both *in vivo* and *in vitro* datasets with statistical significance ($p \leq 0.05$ and $FDR \leq 0.25$). For cell lines, both GSEA and Decipherer

Table 2. Hypothesized post-treatment signaling pathway status in patients

Patient	Response	ERK	Apoptosis	Patient	Response	ERK	Apoptosis
Pt5	P(ositive)	-2.25	1.34	Pt18	No Tumor	-0.15	0.13
Pt8	P	-	0.82	Pt1	N(egative)	0.21	-1.00
Pt9	P	-0.97	-	Pt7	N	-0.10	0.11
Pt14	P	-	-0.86	Pt10	N	1.02	-1.57
Pt16	P	-0.20	1.42	Pt15	N	-	-1.01
Pt17	P	-1.02	1.01	Pt20	N	1.30	-1.68
Pt19	P	-	0.91				

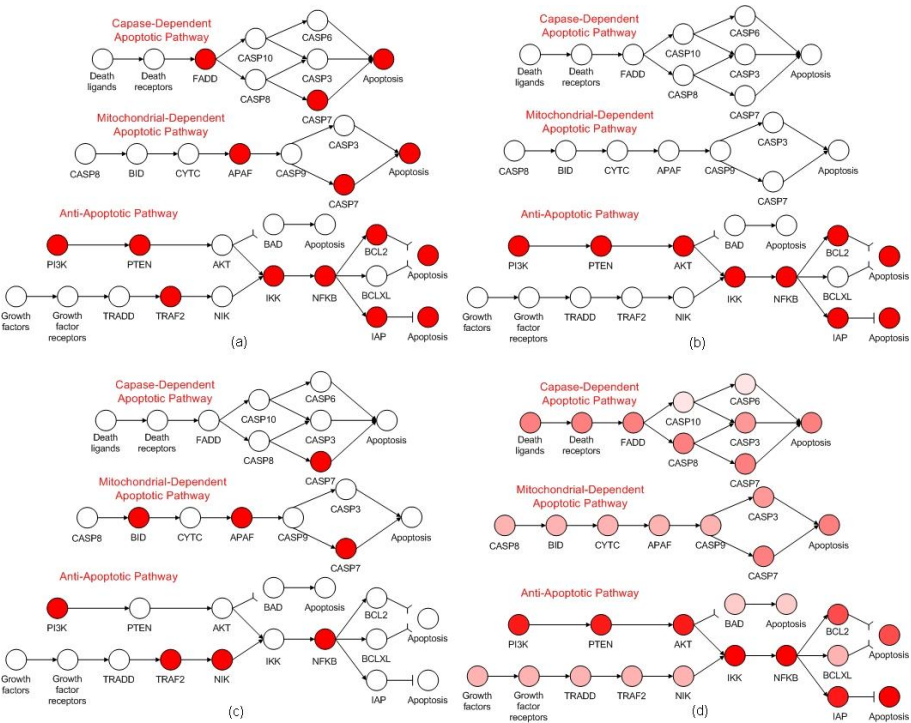


Fig. 4. Contrast results of the genes identified by the leading edge analysis of GSEA and the pathways identified by Decipherer: (a) GSEA applied to the *in vitro* dataset. The identified genes are highlighted. (b) Decipherer applied to the *in vitro* dataset. Since the identifications of CNE1 and HK1 on Apoptosis pathway are the same, this is the only highlighted pathway in the figure. (c) GSEA applied to the *in vivo* dataset. (d) Decipherer applied to the *in vivo* dataset. Color density represents the frequency of a genetic pathway being hypothesized in the patients.

identify the pattern of PI3K-NFκB-IAP pathway (Figure 4a and 4b). However, GSEA misses AKT, and selects some other irrelevant genes. This is because the leading edge analysis ignores relationships between selected genes. When applied to the *in vivo* dataset, GSEA does not identify any strong genetic pathway

pattern (Figure 4c), but for Decipherer, multiple pathways with different significance are identified (Figure 4d). The most significant identification is still PI3K-NF κ B-IAP pathway, which indicates the main genetic response of patients in apoptosis pathway is similar to that of cell lines. Another discovery is the death receptor regulated pro-apoptosis pathway. This pathway is previously undiscovered in the *in vitro* experiments, which means there exists alternative apoptosis regulation pathway for CYC202 in NPC patients rather than in cell lines. Thus, we show that, compared with the leading edge analysis of GSEA, Decipherer generates more biologically meaningful results, which can be used as a guide for further drug research and disease treatment.

3.4 Biological Reasoning and Discussion

Epstein-Barr Virus (EBV) infection dysregulates NF κ B, MAPK, JAK-STAT and PI3K-AKT pathways [21], and thus plays critical role in NPC pathogenesis [12]. Up regulation of NFKB2 and BIRC5 (IAP) induced by EBV increases resistance of NPC cells to apoptosis, which has been confirmed by RNA interference [15]. On the other hand, CYC202 inhibits CDK2, 7 and 9 through competitive inhibition of ATP binding [10]. CDK7 and CDK9 phosphorylate the carboxyl terminal domain of RNA polymerase II, which initiates the gene transcription. NF κ B regulated genes and IAP family members are greatly affected because of their short protein half-life [9]. The suppression of genes in ERK pathway and anti-apoptosis pathway, for example MAPK1, MAPK3, MCL1, BCL2, BIRC4 and BIRC5, are frequently observed in CYC202 treatment [11,22,14,16,8]. In this study, Decipherer identified different regulation of RAS-ERK cell proliferation pathway and PI3K-NF κ B-IAP anti-apoptosis pathway between two outcome groups both *in vitro* and *in vivo*. With the support of literature, we conclude that these two pathways are the main drug pathways of CYC202 in human NPC cells. On the other hand, due to the diversity of individual genetic environment of patients, the hypothesized escape pathways are heterogeneous. The dysregulation of NF κ B pathway and MAPK pathway are both commonly observed in CYC202 resisters. Based on these observations, we made an individual treatment proposal for these CYC202 resisters (See supplementary material for treatment proposal).

4 Conclusion and Future Work

During the past decade, mRNA microarray techniques have been greatly developed and have found many significant applications in biomedical research. In this paper, we present a novel statistical gene expression evaluation framework to discover drug responsive pathways in treatment gene expression data. We decide to report the method, since we have applied Decipherer to two NPC study cases and have found some meaningful results, which have been considered to be applied to improve the CYC202 based NPC treatment in clinic. Thus, we think of Decipherer to be a potential valuable direction to follow. However, we realize

that Decipherer has some caveats. We list following caveats as conclusions of this paper:

- Signaling pathway is a protein level description, while gene expression is an mRNA level measurement. In Decipherer, we use correlation of expression of adjacent genes to score a genetic pathway, which seems to lack enough biological support. Our explanation is that we try to impose a data-driven induction that if we can find a pathway satisfying the criteria, then we have evidence to support the pathway to be interesting. Readers may suspect the correctness of our criteria. We believe that the criteria are still very initial and need to be improved.
- We compare pathways from KEGG pathway database, Pathway Ingenuity database, and WikiPathways database [28]. A surprising observation is that pathways with same pathway name may have maximal approaching 50% disagreement with each other (personal communication with Soh). This observation has questioned our framework since Decipherer is only capable of evaluating known pathways. We believe that some pathway inferences should be included into the future version.
- Metabolic pathways are another important data source for drug action hypothesizing. Patients may response differently to treatment because of their different metabolic rates. For example, two of us recently found, in a colon cancer study, that Fluorouracil, a pyrimidine analog, affected two different metabolic pathways: an effective pathway and a drug degradation pathway. The expression of genes on these two pathways were observed to have strong correlation with treatment outcome [20]. Thus, we believe that metabolic pathways should be considered as well.

Currently, a new version of Decipherer is in development. We have made improvements to overcome some of the caveats listed above:

- A new set of rules has been designed for Decipherer to generate hypotheses. Firstly, a hypothesized pathway should not have broken logic in any responder. For example, if $\langle \text{HRAS}, \text{RAF1}, \text{\$activation} \rangle$ is a relationship for evaluation, we need to exclude the case that HRAS is greatly induced while RAF1 is significantly repressed. Secondly, a hypothesized pathway should be significantly perturbed by treatment in all responders. Thirdly, the regulation of a hypothesized pathway should be consistent in all responders. Finally, a resister should at least violate one of the three criteria above.
- The challenge of taking metabolic pathway into the current design of Decipherer is that metabolic pathways are commonly circular systems, and thus it is difficult to define source and sink genes in a pathway. This issue comes similar with the problem we have mentioned in Section 2.5 that arbitrarily taking source and sink genes in signaling pathways makes the number of hypothesis candidates increase exponentially. To solve the problem, we employ a dynamic procedure to only keep and extend valid candidates for each length, rather than generate all candidates at one time.

Acknowledgement

The NPC patient data are kindly provided by Dr. Boon Cher Goh, National University Hospital Singapore. This work is supported in part by a NUS research scholarship (Dong) and a MOE AcRF Tier 1 grant (Wong).

References

1. Alvi, A., Austen, B., Weston, V., et al.: A novel CDK inhibitor, CYC202 (R-roscovitine), overcomes the defect in p53-independent apoptosis in B-CLL by down-regulation of genes involved in transcription regulation and survival. *al* 105, 4484–4491 (2005)
2. Doniger, S.W., Salomonis, N., Dahlquist, K.D., et al.: MAPPFinder: Using Gene Ontology and GenMAPP to create a global gene-expression profile from microarray data. *Genome Biology* 4(1), R7 (2003)
3. Draghici, S., Khatri, P., Tarca, A., et al.: A systems biology approach for pathway level analysis. *Genome Research* 17(10), 1537–1545 (2007)
4. Guo, Z., Li, Y., Gong, X., et al.: Edge-based scoring and searching method for identifying condition-responsive protein-protein interaction sub-network. *Bioinformatics* 23, 2121–2128 (2007)
5. Herrington, H.: Controlling the false discovery rate in multiple hypothesis testing, <http://www.unt.edu/benchmarks/archives/2002/april02/rss.htm>
6. Ideker, T., Ozier, O., Schwikowski, B., Siegel, A.F.: Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics* 18(suppl. 1), S233–S240 (2002)
7. Kanehisa, M., Goto, S., Kawashima, S., Nakaya, A.: The KEGG database at GenomeNet. *Nucleic Acids Research* 30(1), 42–46 (2002)
8. Lacrima, K., Valentini, A., Lambertini, C., et al.: In vitro activity of cyclin-dependent kinase inhibitor CYC202 (Seliciclib, R-roscovitine) in mantle cell lymphomas. *Annals of Oncology* 16, 1169–1176 (2005)
9. Lam, L., Pickeral, O., Peng, A., et al.: Genomic-scale measurement of mRNA turnover and the mechanisms of action of the anti-cancer drug flavopiridol. *Genome Biology* 2(10), research004 (2001)
10. McClue, S., Blake, D., Clarke, R., et al.: In vitro and *in vivo* antitumor properties of the cyclin dependent kinase inhibitor CYC202 (R-ROSCOVITINE). *International Journal of Cancer* 102, 463–468 (2002)
11. Meijer, L., Borgne, A., Mulner, O., et al.: Biochemical and cellular effects of roscovitine, a potent and selective inhibitor of the cyclin-dependent kinase cdc2, cdk2 and cdk5. *European Journal of Biochemistry* 243, 527–536 (1997)
12. Pathmanathan, R., Prasad, U., Sadler, R., et al.: Clonal proliferations of cells infected with Epstein-Barr virus in preinvasive lesions related to nasopharyngeal carcinoma. *The New England Journal of Medicine* 333, 693–698 (1995)
13. Pui, C., Evans, W.: Acute lymphoblastic leukemia. *New England Journal of Medicine* 339, 605–615 (1998)
14. Raje, N., Kumar, S., Hideshima, T., et al.: Seliciclib (CYC202 or R-roscovitine), a small-molecule cyclin-dependent kinase inhibitor, mediates activity via down-regulation of MCL1 in multiple myeloma. *Blood* 106, 1042–1047 (2005)
15. Shi, W., Bastianutto, C., Li, A., et al.: Multiple dysregulated pathways in nasopharyngeal carcinoma revealed by gene expression profiling. *International Journal of Cancer* 119, 2467–2475 (2006)

16. Smith, P., Yue, E.: *Inhibitors of Cyclin-dependent Kinases as Anti-tumor Agents*. Taylor and Francis Group, Abington (2006)
17. Soh, D., Dong, D., Guo, Y., Wong, L.: Enabling more sophisticated gene expression analysis for understanding diseases and optimizing treatments. *ACM SIGKDD Explorations* 9, 3–14 (2007)
18. Subramanian, A., Tamayo, P., Mootha, V., et al.: Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Science of the United States of America* 102, 15545–15550 (2005)
19. Sultanem, K., Shu, H.K., Xia, P., et al.: Three-dimensional intensity-modulated radiotherapy in the treatment of nasopharyngeal carcinoma: the University of California, San Francisco experience. *International journal of radiation oncology, biology, physics* 48, 711–722 (2000)
20. Tan, W., Dong, D., Loh, M., et al.: Pathway determinants of 5-Fluorouracil activity. Poster. In: 20th EORTC-NCI-AACR symposium on Molecular targets and Cancer Therapeutics (2008)
21. Tsao, S., Tramoutanis, G., Dawson, C., et al.: The significance of LMP1 expression in nasopharyngeal carcinoma. *Cancer Biology* 12, 473–487 (2002)
22. Whittaker, S., Walton, M., Garrett, M., Workman, P.: The cyclin-dependent kinase inhibitor CYC202 (R-Roscovitine) inhibits retinoblastoma protein phosphorylation, causes loss of cyclin D1, and activates the mitogen-activated protein kinase pathway. *Cancer Research* 64, 262–272 (2004)
23. Yeoh, E., Ross, M., Shurtleff, S., et al.: Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell* 1, 133–143 (2002)
24. Yu, M., Yuan, J.: Epidemiology of nasopharyngeal carcinoma. *Seminars in Cancer Biology* 12, 421–429 (2002)
25. Zeeberg, B.R., Feng, W., Wang, G., et al.: GoMiner: A resource for biological interpretation of genomic and proteomic data. *Genome Biology* 4(4), R28 (2003)
26. Zien, A., Kuffner, R., Zimmer, R., Lengauer, T.: Analysis of gene expression data with pathway scores. *Proceedings of International Conference on Intelligent Systems for Molecular Biology* 8, 407–417 (2000)
27. Pathway Ingenuity database, <http://www.ingenuity.com/>
28. WikiPathways, <http://www.wikipathways.org/index.php/WikiPathways>

Selection of Multiple SNPs in Case-Control Association Study Using a Discretized Network Flow Approach *

Shantanu Dutt^{1,**}, Yang Dai^{2,**}, Huan Ren¹, and Joel Fontanarosa²

¹ Department of Electrical and Computer Engineering

² Department of Bioengineering

SEO, 851 South Morgan Street, Chicago, IL 60607, USA

Abstract. Recent large scale genome-wide association studies have been considered to hold promise for unraveling the genetic etiology of complex diseases. It becomes possible now to use these data to assess the influence of interactions from multiple SNPs on a disease. In this paper we formulate the multiple SNP selection problem for determining genetic risk profiles of certain diseases by formulating novel 0/1 IP formulations for this problem, and solving them using a new near-optimal and efficient discrete optimization technique called *discretized network flow* that has recently been developed by us. One of the highlights of our approach to solving the multiple SNP selection problem is recognizing that there could be different genetic profiles of a disease among the patient population, and it is thus desirable to classify/cluster patients with similar genetic profiles of the disease while simultaneously selecting the right genetic marker sets of the disease for each cluster. This approach coupled with the DNF technique has yielded results for several diseases with some of the highest sensitivities seen so far and specificities that are higher or comparable to state-of-the-art techniques, at a fraction of the runtime of these techniques.

Keywords: Multiple SNPs, Case-Control Study, Optimization, Discretized Network Flow.

1 Introduction

Recent large-scale, high-density genome-wide association (GWA) studies have improved our understanding of the genetic basis of many complex traits. Various published associations have not been replicated in comparable GWA studies, possibly due in part to the omission of interactions among disease-associated loci (called *epistasis*) from many statistical models [13], [9]. Increasing empirical evidence suggests that interactions among loci contribute broadly to complex human diseases. The task in epistatic study is identification of a set of k single nucleotide polymorphisms (SNPs) and the corresponding allele types that are associated with the disease. We call it the *k-SNP marker selection problem*. Much of the recent statistical work has focused on interaction models that have

* This work was supported in part by NSF grants CCR-0204097 and CCF-0811855.

** Corresponding authors.

small or no marginal effects at each locus. Since the number of possible interaction combinations among the genotyped markers is astronomical for a large scale case-control association study, it is prohibitive to search one or a very few disease-related interactions among all these combinations. Several methods based on brute-force search have been developed, including the combinatorial partitioning method (CPM) [15], and multifactor-dimensionality reduction (MDR) [17]. While these techniques have been used to effectively analyze data sets of small scale, they cannot be scaled-up for large data sets. More methods can be found in a review paper [14]. A result based on a Bayesian statistical model has indicated feasibility of genome scale epistatic analysis [20]. The model was applied to an association study set with 96,932 markers genotyped from 146 individuals (96 affected and 50 controls) for 2-SNP and 3-SNP mark set selections. The run time was about 5 hours on a Pentium M 1.6GHz laptop with 512 Mb memory.

Brinza and co-authors considered the problems of searching for the most disease-associated and the most disease-resistant k -SNP marker sets [3, 4]. Combinatorial methods, such as greedy algorithms, were proposed to search these k -SNP marker sets. Using the selected marker set the authors further considered the disease susceptibility prediction problem, i.e., predict an individual's disease status based on the marker set. The algorithms have been shown to outperform other machine learning methods on three small-scale data sets. Recently, the above approach was further extended to searching for a k -SNP marker set which has the best odds ratio, a criterion often used in disease association studies [5].

In this paper, we propose a novel optimization-based approach to the k -SNP marker selection problem for large-scale SNP data. The detection of k -SNP mark set is formulated as 0/1 integer programming problems. Our formulation can simultaneously discover subgroups of cases and their corresponding best marker sets. The core technique for fast near-optimal solutions of the 0/1 IP problem is a recently developed new methodology called *discretized network flow* (DNF). DNF is a general computing framework for obtaining high-quality discrete optimization problems (DOPs) solutions in tractable run-times. The efficacy of DNF has been established in the realm of VLSI CAD for many hard DOPs [7, 8, 16]. DNF combines the computational efficiency of continuous optimization methods, in that it uses network flow, an optimal continuous optimization technique, as its core algorithmic process, with novel discretization techniques so that near-optimal legal discrete solutions are efficiently obtained.

The proposed approaches to epistasis analysis in GWAs SNP data sets are based on novel 0/1 IP formulations of multi-locus marker detection and use DNF to solve these formulations. The effectiveness of the approach are evaluated using 5 data sets in a previous study [5]. The new proposed method significantly outperformed previous methods in most cases: sensitivity¹ and specificity² in a 5-fold cross-validation test increased by 81% and 38.8%, respectively, from the results of MDR, and by 14% and -9.2% (the negative value indicates deterioration), respectively, from the results of the combinatorial method CPS [4], with a runtime that is a fraction of the runtimes of these methods.

¹ The percentage of cases correctly identified with the selected SNP marker set.

² The percentage of controls correctly identified with the selected SNP marker set.

2 Method

2.1 0/1 Integer Programming (IP) Formulations

We describe here our 0/1 IP models for optimizing the selection of a set of SNPs and their associated allele so that the chosen SNP-allele pairs are strong distinguishing markers between case and control. We consider un-phased genotype data of an experiment involving m_1 and m_2 individuals in case and control groups respectively. For each SNP there are 3 allele types. Given a pair $p_{i,j}$ of SNP i and allele j (henceforth we will use the terms ‘‘SNP-allele pair’’ or just ‘‘allele’’ to refer to a $p_{i,j}$), we define $c_{i,j}(x) = 1$ if the x 'th individual P_x in the case group has allele j at SNP i , and $c_{i,j}(x) = 0$ otherwise. Similarly, we also define $h_{i,j}(z) = 1$ if the z 'th individual NP_z in the control group has allele j at SNP i , and $h_{i,j}(z) = 0$ otherwise. The presence of allele $p_{i,j}$ in an individual is denoted by marker $p_{i,j}^1$, while its absence is denoted by marker $p_{i,j}^0$.

We define the per-case benefit $b_{i,j}(x)$ of a SNP-allele pair $p_{i,j}$ for an individual x as:

$$b_{i,j}(x) = \left| c_{i,j}(x) - \frac{\sum_{z=1}^{m_2} h_{i,j}(z)}{m_2} \right| \tag{1}$$

$b_{i,j}(x)$ is a good indicator of discriminative ability between case individual x and the control group for allele j at SNP i . Furthermore, $b_{i,j}(x)$ is also a correct indicator of the specificity of allele $p_{i,j}$ as we show below.

Claim 1 *The $b_{i,j}(x)$ definition is consistent with the specificity of allele $p_{i,j}$.*

Proof: There are two cases.

Case 1: $c_{i,j}(x) = 1$, i.e., $p_{i,j}$ is present in P_x . Then the second term in Eqn. □ is the fraction of controls that also contain $p_{i,j}$. Thus higher $b_{i,j}(x)$ is, lower is this fraction, which means high specificity for $p_{i,j}$.

Case 2: $c_{i,j}(x) = 0$, i.e., $p_{i,j}$ is absent in P_x . The second term in Eqn. □ can be re-stated as: [1 – fraction of controls in which $p_{i,j}$ is absent], which is also the definition of $b_{i,j}(x)$, since $c_{i,j}(x) = 0$. Thus higher $b_{i,j}(x)$ is, smaller is the fraction of controls for the $p_{i,j}^0$ marker, and thus higher is the specificity for this marker.

Note that in each case, $b_{i,j}(x)$ is, in fact, exactly the specificity of $p_{i,j}^{c_{i,j}(x)}$. \diamond

A good target set of SNP-allele combinations for the entire case group can be one in which the selected SNP-allele pairs $p_{i,j}$ s have the same value of $c_{i,j}()$ for all case individuals and whose sum of benefits is the maximum (greatest distinction from the controls). We thus formulate the following 0/1 integer programming (IP) for the k -SNP selection problem.

We first define the benefit-based similarity metric $s(x, y, i, j, val)$ between two individuals in the case group P_x and P_y for a SNP-allele pair marker $p_{i,j}^{val}$ ($val \in \{0, 1\}$) as:

$$s(x, y, i, j, val) = \begin{cases} (b_{i,j}(x))^\alpha + (b_{i,j}(y))^\alpha & \text{if } c_{i,j}(x) = c_{i,j}(y) = val \\ -\infty & \text{otherwise.} \end{cases} \tag{2}$$

The α parameter above magnifies (when $\alpha > 1$) or shrinks (when $\alpha < 1$) the ratio of benefits of different alleles. This is useful when there are constraints on the selection of alleles and $\alpha > 1$ can be used to give a magnified priority to the

selection of alleles with high benefit. Further, let $d_{i,j}(val)$ be a 0/1 variable which indicates if $p_{i,j}^{val}$ is selected as a marker in the SNP-allele set ($d_{i,j}(val) = 1$) or not ($d_{i,j}(val) = 0$). The 0/1 IP formulation for the problem of the maximum-benefit SNP/allele set election is:

$$\begin{aligned}
 &\text{Maximize : } \sum_{p_{i,j}^{val}} \sum_{1 \leq x \leq m_1} \sum_{1 \leq y \leq m_1, y \neq x} s(x, y, i, j, val) \cdot d_{i,j}(val) \\
 &\text{Subject to : } (i) \sum_{j=1}^3 d_{i,j}(0) + d_{i,j}(1) \leq 1, \forall \text{ SNPs } i. \\
 &\quad (ii) \sum_{p_{i,j}} (d_{i,j}(0) + d_{i,j}(1)) \leq k.
 \end{aligned} \tag{3}$$

It is easy to see from the definition of $s(x, y, i, j, val)$ ($-\infty$ value for mismatched alleles between a pair of individuals in the case group), that in order to maximize the objective function, no SNP-allele pairs $p_{i,j}$ will be selected that differ in $c_{i,j}$ value among any pair of individuals in the case group. Thus only common SNP-alleles across all individuals in the group will be selected.

Simultaneous Patient Clustering and k -SNP Marker Selection. One issue that complicates the marker selection problem is that the genetic reasons of a disease for patients with different ethnic backgrounds can be different, and it can also be different within each ethnic group. Therefore, in order to improve the accuracy of selected markers, it is desirable to incorporate a clustering/classification step for individuals of the case group simultaneously with the marker selection process so that clusters are automatically formed based on similarities of genetic disease markers. Different markers can thus be selected for different clusters to best match their genetic disease profile. The IP problem given in Eqn. 3 put all individuals in the case group in one cluster and tries to find the best marker for this cluster, which may not be very strong since we are forcing marker selection into patients with potentially different genetic disease profiles. The formulation below is for partitioning individuals in the case group into up to G clusters, each with similar genetic disease profiles and simultaneously finding their best SNP-allele markers.

$$\begin{aligned}
 &\text{Maximize : } \sum_{1 \leq g \leq G} \sum_{p_{i,j}^{val}} \sum_{1 \leq x \leq m_1} \sum_{1 \leq y \leq m_1} s(x, y, i, j, val) \cdot b_x^g \cdot b_y^g \cdot d_{i,j}^g(val) \\
 &\text{Subject to : } (i) \sum_{j=1}^3 d_{i,j}^g(0) + d_{i,j}^g(1) \leq 1, \forall \text{ SNPs } i \text{ and } \forall \text{ clusters } g. \\
 &\quad (ii) \sum_{1 \leq g \leq G} b_x^g = 1, \forall x. \quad (iii) \sum_{p_{i,j}} (d_{i,j}^g(0) + d_{i,j}^g(1)) \leq k, \forall g.
 \end{aligned} \tag{4}$$

where G is an upper bound on the number of patient groups/clusters, b_x^g is a 0/1 variable that is 1 if P_x is chosen to be in cluster g , and 0 otherwise, $d_{i,j}^g(val)$ is a 0/1 variable that is 1 if $p_{i,j}^{val}$ is chosen as a SNP-allele marker for cluster g and is 0 otherwise, and k is an upper bound on the number of $p_{i,j}$'s selected in the marker set for each cluster. We note that an individual can belong to only one cluster, while $p_{i,j}^{val}$ may be chosen to be in the SNP-allele marker set for multiple clusters (both $p_{i,j}^0$ and $p_{i,j}^1$ of course cannot be in the same marker set).

The optimization problems formulated above involve large numbers of 0/1 integer variables that are indicative of problems which even state-of-the art MIP solvers (e.g. CPLEX) could fail to solve within a reasonable time frame. We propose network flow formulations of the problems, which can be solved approximately and efficiently by using the discretized network flow method.

2.2 The Discretized Network Flow Technique

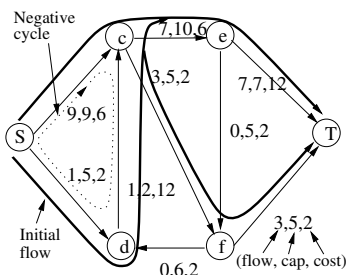


Fig. 1. A feasible flow for sending 10 units of flow from source (S) to sink (T) is shown by curved dark lines. Arc labels are arranged as (flow, capacity, cost).

The class of problems to which discretized network flow (DNF) can be applied are DOPs that can be modeled as mixed integer programming (MIP) problems with linear and non-linear (polynomial, min and max) objective functions and constraints; these encompass a very large class of DOPs. The basic idea of this technique can be described as follows.

A network flow graph G is a directed graph in which each arc e has a capacity $cap(e)$ and a cost $cost(e)$; see Fig. 1. The capacity of an arc indicates the maximum amount of flow that can pass through the arc; the cost of an arc is the cost incurred per unit flow through the arc. The *minimum cost flow problem* in G is

to find a way to pass a certain amount of flow through G from a source node S to a sink node T that has minimum cost. Network flow is an elegant continuous optimization technique that has found applications in many domains [1].

In standard network flow [1], the flow through an arc can be any continuous (i.e., real) value, and if there is incoming flow into a node u , there can be outgoing flow into any subset of outgoing arcs from u . In order to solve DOPs using a network flow approach (due to its time efficiency), we need flows in the graph G to have certain discrete properties. Briefly, the two most important of these are:

(1) *Discrete Arcs* : Some arcs e will need to have a “binary-valued” discrete cost and flow amount structure, i.e., such a *discrete arc* can only have two (flow amount, cost incurred) pairs: $\{(0, 0), (cap(e), cost(e))\}$. This means that any initial flow amount f through e will incur a cost of $cost(e)$ (irrespective of f), and, if $f < cap(e)$, then any subsequent flow through e will incur 0 cost. Such arcs are called *discrete arcs*, and their cost structure makes the incurred cost a concave function of the flow amount f . We thus use the near-optimal concave min-cost flow algorithm of [10] in all our network flow computations.

(2) *Mutual Exclusiveness* : For some nodes u at most one of their output arcs and/or one of their input arcs can have flow in them; see Fig. 2. We call this the *mutually exclusive output arcs (MEA)* requirement.

Of these, the MEA requirement is central in the solution of the 0/1 IPs for multiple allele selection discussed in Subsec. 2.1. Our DNF technique uses special algorithms and arc cost formulations using non-objective-function based costs (in addition to, of course, objective-function based costs) in order to achieve the above types of discretizations without sacrificing much in optimality [7, 8, 16]. We briefly discuss below our MEA satisfaction technique.

Satisfying MEA Constraints in DNF. In general, the cost of arcs in a n/w graph G is determined based on the objective function being minimized. However, for the purpose of MEA satisfaction (and, in fact, for some other flow constraints as well), for each arc e in an MEA set, we add to its function based cost $C(e)$, a discrete *base cost* $C'(e)$, which is a constant for all edges e to which it is applied; thus its total cost $cost(e) = C'(e) + C(e)$. For various flow constraints,

including MEA, that we considered, an invalid flow will always incur at least an extra C' cost. The $C'(e)$ cost is thus kept large enough so that a min-cost invalid flow always incurs more total cost than a valid min-cost flow (even though the objective-function part of this cost, the C-cost, incurred by an invalid flow may be less than that incurred by a valid flow) [16]. The correctness of this technique for MEA satisfaction has been established in [16], and is re-stated below using the terminology of this paper.

Theorem 1. [16] *Any min-cost flow in G with added C' costs on MEA arcs will satisfy all MEA constraints.*

2.3 Application of DNF to k -SNP Selection

In our DNF model, we approximate the 0/1 IP of Eqn. 4 by obtaining all clusters via simultaneous recursive bi-partitioning and allele marker set selections. We start with the set of all cases as the initial single cluster C_1 , and then recursively bi-partition each cluster C_i at the current level of bi-partitioning into two clusters $C_{i,1}$ and $C_{i,2}$, and simultaneously select their allele marker sets. The process continues as long as the overall specificity, by so doing, increases, and the upper bound of G clusters is not violated³.

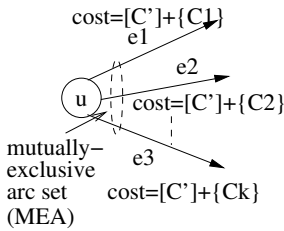


Fig. 2. An MEA set and DNF structure for satisfying MEA requirements. The C' costs shown are discrete, indicated by square brackets; the other costs indicated by curly brackets are continuous.

The network flow model for the bi-partition plus k -SNP selection problem is shown in Fig. 3. For each cluster C_i , the network flow graph G_i consists of two subgraphs $G_{i,1}, G_{i,2}$ corresponding to the two clusters to be formed, as shown in Fig. 3(a). In each subgraph, there are two levels of duplicated case “meta nodes” for each case individual. These two levels in each $G_{i,r}$ ($r = 1, 2$) are connected by a complete meta bipartite graph, where a “meta edge” connects each pair of meta nodes P_x, P_y . As shown in Fig. 3(b), each meta edge is a collection of allele-to-allele connections between alleles of P_x and P_y with the same allele values (0 or 1) that have costs $-s(x, y, i, j, val)$. A case P_x is selected to be in $C_{i,r}$ if there is flow through its two meta nodes in the two levels of

$G_{i,r}$. An allele $p_{i,j}$ is selected as a marker in $C_{i,r}$ if flow passes through the $p_{i,j}$ nodes in $G_{i,r}$ corresponding to each P_x selected in $C_{i,r}$ (we shortly show that if a min-cost flow passes through any $p_{i,j}$ node of any selected P_x , it will pass through all $p_{i,j}$ nodes of all selected P_x 's in a cluster; this implies that a common set of (up to k) alleles will be selected for each selected P_x in a cluster, and hence a correct allele marker set is selected for each cluster). The min-cost flow⁴ through subgraphs $G_{i,1}$ and $G_{i,2}$ will select those P_x 's in each

³ In our DNF modeling, we have not explicitly set an upper bound on the number of clusters; however, the number of clusters determined naturally turns out to be a small constant in the range [6, 16].

⁴ We solve the maximization problem of Eqn. 4 by using min-cost network flow by setting the cost of arcs for $p_{i,j}$ selection among node pairs P_x, P_y to be the negative of $s(x, y, i, j, val)$, which is the contribution of this selection to the maximization objective of Eqn. 4.

subgraph that minimizes the total cost of all selected allele-to-allele arcs among all the selected P_x 's (this corresponds to maximizing the objective function of Eqn. 4).

In order for the flow to partition C_i into two valid clusters with valid allele marker selections for them, two sets of constraints need to be satisfied: (1) each P_x is selected in only one of $C_{i,1}$, and $C_{i,2}$. (2) In each $G_{i,r}$, the flow goes through the same set of allele nodes in each P_x meta node that the flow selects.

The first constraint is satisfied via MEA requirements on input arcs to case meta nodes in the first level of each $G_{i,r}$, and on the outgoing arcs from case meta nodes in the second level of each subgraph; see Fig. 3. This causes the flow to pass through P_x meta nodes in only one subgraph.

Fig. 3(b) shows the detailed structure of the meta-node of a case P_x , as well as the details of connection between the meta nodes of cases P_x and P_y in the two levels. The structure in each individual meta-node is a tree with allele nodes for each SNP at the leaf level. Let m be the number of cases in C_i to be partitioned. An incoming flow of amount km to each meta-node in the 1st level is distributed through the tree structure to k leaf allele nodes (this indicates that the corresponding k SNP-allele pairs are selected as markers). Note that the incoming arcs to leaf allele nodes are discrete arcs with capacity m and objective function independent cost C' . Hence, to minimize total cost, flow is only distributed on k such arcs with full flow of amount m on each arc, i.e., to k leaf allele nodes.

The meta arcs between P_x meta nodes are each a set of connections between corresponding alleles, i.e., the SNP-allele pair $p_{i,j}$ present in the P_x meta-node is connected to that in the P_y meta node if and only if $c_{i,j}(x) = c_{i,j}(y)$; the cost of this arc is $-s(x, y, i, j, val)$. When both P_x and P_y are selected in one cluster, the allele to allele connections make sure that the flow passes through (i.e., select) the same k alleles as markers for the two cases. The cost of the arcs between the two P_x meta nodes in the two levels of a subgraph is $-\infty$. Hence, if a meta node P_x has a flow through it in level 1 (meaning it is selected to be in the corresponding cluster), then part of this flow will also go through the level-2 P_x meta node in that subgraph; the allele markers selected by the flow are also the same in both P_x meta nodes in the two levels.

The second constraint mentioned above is satisfied as follows. Due to the complete bipartite connection between case meta nodes at the two levels of each subgraph, all cases selected in a cluster subgraph will also select the same k alleles as markers. Let us assume that this is not the case, so that the sets of k alleles selected for two meta nodes P_x, P_y that are selected in level 1 to be in the same cluster are different. Consider the P_y meta node in level 2. The flow through the k alleles of P_x will be forced into the same k alleles of the 2nd-level P_y via the allele-to-allele connections, and the flow through each allele incurs a discrete C' cost. The flow from the k alleles of the 1st-level P_y will also be forced through the corresponding alleles in the 2nd-level P_y . Since the two sets of k alleles are not the same, there will be flow through at least $(k + 1)$ alleles in the 2nd level P_y . This thus incurs $(k + 1) C'$ costs, instead of only $k C'$ costs for a valid flow—in which a consistent set of k alleles are selected for all meta-nodes in the same cluster. Thus the invalid flow that goes through at least $(k + 1)$ alleles in the 2nd-level P_y meta-node cannot be a min-cost flow; in other words,

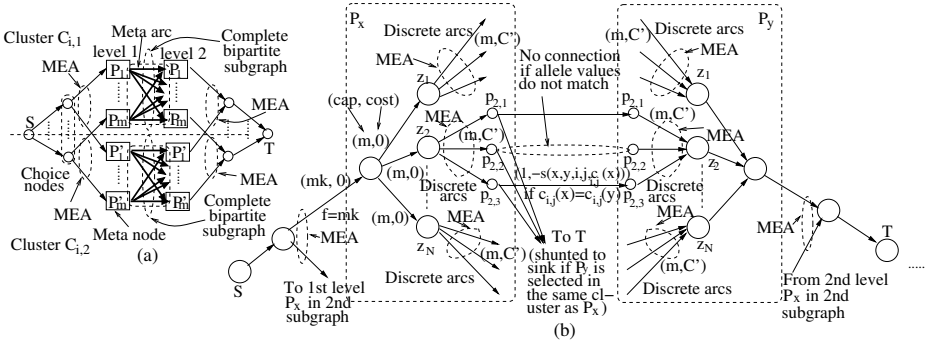


Fig. 3. (a) High level network flow model for the bipartition problem. (b) Detailed patient meta node structure, and connection between patient meta nodes.

a min-cost flow will always select a common set of k alleles across all selected meta-nodes in each subgraph.

2.4 Improved 0/1 IP Formulation and DNF Model with Explicit Specificity Consideration

As explained in Sec. 2.1, the formulation of Eqn. 4 is geared toward choosing a common set of allele markers for each cluster, which implies a sensitivity of 1 for its cluster. This in turn implies a high sensitivity for the entire set of allele marker sets (one marker set per cluster)⁵. As established in Claim 11, the definition of $b_{i,j}(x)$ is its specificity. Thus the selection of high-benefit markers (as per the objective function of Eqn 4) is conducive to high specificity for each allele marker set M_i , and thus also to high specificity for the set $\mathcal{M} = M_1, \dots, M_G$ of allele marker sets, assuming a uniform distribution of mismatching controls across alleles. For example, consider a 2-element allele marker set $M_i = (p_{i1,j1}^0, p_{i2,j2}^1)$, and let each allele marker have a specificity of 0.6, i.e., the probability of a control mismatching each allele marker is 0.6. Assuming an uniform distribution of controls that mismatch each allele marker, the probability of a control mismatching marker set M_i is the probability that it mismatches either allele marker = 1 - (the probability that it matches both allele markers) = $1 - (1 - 0.6)^2 = 0.84$. However, the mismatching controls may not be uniformly distributed, and if there is some clustering of the controls that mismatch the alleles of M_i , then the specificity of M_i could be much lower than 0.84. For example, if the set of controls that mismatch both allele markers are the same, then the specificity of M_i is only 0.6. Thus a more direct method is needed for identifying allele markers so that the specificity of each M_i and that of \mathcal{M} is high. Toward that end, we formulate a modified 0/1 IP and then discuss its DNF model below.

⁵ For a case-clustered solution with multiple allele marker sets $\mathcal{M} = M_1, \dots, M_G$, each marker set M_i corresponding to cluster C_i , a case has a match with the set of marker solutions \mathcal{M} if it has a match with *any* allele marker set M_i ; it then contributes to the sensitivity of \mathcal{M} (i.e., to the true positive [TP] number; see Eqn. 7). Conversely, a control has a mismatch with \mathcal{M} if the individual has a mismatch with *all* marker sets in \mathcal{M} , and only then does it contribute to the specificity of \mathcal{M} (i.e., to the true negative [TN] number; see Eqn. 7).

We first define the function $dis(z, i, j)$ between a control individual NP_z and all case individuals that mismatch the allele value of $p_{i,j}$ in NP_z as:

$$dis(z, i, j) = k \cdot \frac{m_1}{m_2} \cdot m_{avg} \cdot Avg_{\{(P_x | c_{i,j}(x) \neq c_{i,j}(z)\}} 2(b_{i,j}(x))^\alpha \quad (5)$$

where Avg_S is the average function over the set S and m_{avg} is the average number of cases in a cluster. The above function is the benefit associated with NP_z mismatching the $p_{i,j}$ value present in a subset of cases, and is tuned so that the total benefit associated with a unit percentage contribution to specificity (due to controls mismatching alleles selected in marker sets) is approximately equal to the total benefit (see objective functions of Eqns. [4](#) and [6](#)) associated with a unit percentage contribution to sensitivity.

Let $\{C_g\}$ be the set of clusters, and $d_{i,j}^g(*)$ be a 0/1 variable that is 1 if and only if $p_{i,j}$ is not selected as an allele marker in cluster C_g . Then a control NP_z does not match any $p_{i,j}$ selection in any of the allele marker sets $\{M_g\}$ if $\prod_{g=1}^G [d_{i,j}^g(not((c_{i,j}(z)))) + d_{i,j}^g(*)] = 1$. We thus include these terms for each $p_{i,j}$ in the maximization objective function of the new 0/1 IP given below.

$$\begin{aligned} \text{Maximize : } & \sum_{1 \leq g \leq G} \sum_{p_{i,j}^{val}} \sum_{1 \leq x \leq m_1} \sum_{1 \leq y \leq m_1} s(x, y, i, j, val) \cdot b_x^g \cdot b_y^g \cdot d_{i,j}^g(val) \\ & + \sum_{p_{i,j}} \sum_{1 \leq z \leq m_2} \prod_{g=1}^G [d_{i,j}^g(not((c_{i,j}(z)))) + d_{i,j}^g(*)] \cdot dis(z, i, j) \\ \text{Subject to : } & (i) \sum_{j=1}^3 d_{i,j}^g(0) + d_{i,j}^g(1) + d_{i,j}^g(*) = 1, \forall \text{ SNPs } i \text{ and } \forall \text{ clusters } g. \\ & (ii) \sum_{1 \leq g \leq G} b_x^g = 1, \forall x. \quad (iii) \sum_{p_{i,j}} (d_{i,j}^g(0) + d_{i,j}^g(1)) \leq k, \forall g. \end{aligned} \quad (6)$$

The term $\prod_{g=1}^G [d_{i,j}^g(not((c_{i,j}(z)))) + d_{i,j}^g(*)] \cdot dis(z, i, j)$ in the objective function forces a selection of alleles $p_{i,j}$ in marker sets of clusters that most controls will mismatch in their values (recall that value=1 indicates presence of $p_{i,j}$ and value=0 indicates absence). The corresponding DNF structure corresponding to this term for each $p_{i,j}$ are two chain structures, one for marker $p_{i,j}^1$ called the 1-chain and the other for marker $p_{i,j}^0$ called the 0-chain. Each chain has G sequentially connected ‘‘gateway’’ subgraphs SG_g ’s through which a flow amount of m_2 coming from the source S can potentially pass; see Fig. [4](#). Each SG_g is controlled by a flow coming into it from the network subgraph in which cluster C_g is being formed. Each SG_g has a 0-node and a 1-node, and if the chain is a 1-chain, and either $p_{i,j}^1$ is selected as a marker for C_g or no $p_{i,j}$ (i.e., neither $p_{i,j}^1$ nor $p_{i,j}^0$) is selected as its marker, a unit control flow will come in from C_g into the 1-node of SG_g . There it incurs a C' cost on the arc leading to SG_{t+1} , which means a min-cost flow will choose to push the flow from S along this arc and into SG_{t+1} . If, however, $p_{i,j}^0$ is selected as a marker for C_g , then the control flow from C_g goes into the 0-node of SG_g , which incurs a C' cost in the shunting arc of SG_g that leads to the sink T . This means that a min-cost flow will choose to divert the flow from S to this shunting arc and the flow thus does not go through the 1-nodes of this 1-chain, and is finally shunted to the sink as shown in Fig. [4](#). The structure for a 0-chain for $p_{i,j}$ is analogous. It is thus clear that the flow from S will pass through this chain iff the term $\prod_{g=1}^G [d_{i,j}^g(not((c_{i,j}(z)))) + d_{i,j}^g(*)] = 1$.

If flow reaches the end of, say, a 1-chain for $p_{i,j}$, it is then diverted to a structure for each NP_z that has marker $p_{i,j}^0$ where it incurs a $-dis(z, i, j)$ cost. Thus a min-cost flow choosing such a path means that the corresponding NP_z ’s through which the flow finally passes do not match the $p_{i,j}$ marker selection, if

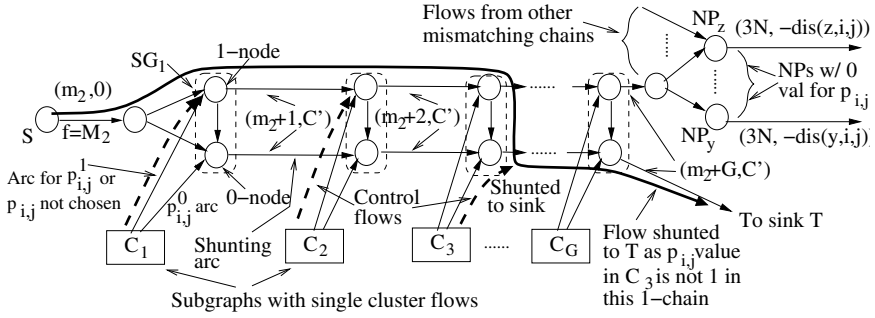


Fig. 4. DNF model for computing the product of 0/1 variables: a 1-chain for allele $p_{i,j}$. The main flow (from S) is shown by a dark solid curve, and control flows are shown by dark dashed curves.

any, in any of the clusters. This implies a contribution toward high specificity of the set of marker sets $\{M_{i,j}\}$. 1- and 0-chains for $p_{i,j}$ supplement the structure of Fig. 3 to solve the 0/1 IP formulation of Eqn. 6. Compared to solving the formulation of Eqn. 4, the solutions to the formulation of Eqn. 6 provided an 11.5% average improvement in specificity with an 8.4% average deterioration in sensitivity, for the data sets for the five diseases discussed in Sec. 3.

3 Computational Study

3.1 Data Sets

We consider 5 data sets which were used in 5 for the evaluation of our method:

- 1) Crohn’s disease: This data set consists of genotypes of 103 SNPs from 144 Crohn’s disease patients and 243 healthy controls. The 103 SNPs locate on a 616 KB region of human Chromosome 5q31 that may contain a genetic variant responsible for the disease. The cases and controls are individuals from 129 trios 6.
- 2) Autoimmune disorder: This data set consists of genotypes of 108 SNPs from 384 cases of autoimmune disorder and 652 controls. The SNPs are selected from a 330KB of human CAN containing gene CD28, CTLA4 and ICONS, which are proved related to autoimmune disorder 19.
- 3) Tick-borne encephalitis: This data set consists of genotypes of 41 SNPs of 21 patients with tick-borne encephalitis virus and 54 patients with mild disease 3.
- 4) Rheumatoid arthritis: This data set consists of genotypes of 2300 SNPs from 460 patients with rheumatoid arthritis and 460 controls. The SNPs are selected by Illumina for an approximately 10KB region of chromosome 18q that showed evidence for linkage in the U.S. and French linkage scans 3.
- 5) Lung cancer: This data set consists of genotypes of 141 SNPs from 322 German male smokers with lung cancer and 273 age-matched healthy smokers. The 141 SNPs are selected from a total of 83,715 SNPs that had been screened using genome-wide DNA pooling strategy, because they showed putative allelic imbalance between case and control DNA pools 18.

3.2 Results

We use the 5-fold cross-validation procedure to evaluate our method. The criteria for evaluation of the predictive ability of the algorithm are sensitivity (Sens) and specificity (Spec). Sens and Spec are defined as follows.

$$\text{Sens} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad \text{and} \quad \text{Spec} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{7}$$

where TP, FP, TN and FN are the numbers of true positives, false positives, true negatives and false negatives, respectively, from the 5-fold CV test (see footnote 4, p. 7, for an explanation of how these numbers are determined for our clustering-based multiple marker sets).

Table 1. Sensitivity, specificity and runtimes of the 5-fold CV procedure using the DNF model to solve the 0/1 IP given in Eqn. 6

Data set	$k = 4, \alpha = 1.5$				$k = 5, \alpha = 1.5$			
	# Clusters	Sens (%)	Spec (%)	Runtime (sec)	# Clusters	Sens (%)	Spec (%)	Runtime (sec)
Autoimmune disorder	11	71.1	57.1	2941	12	84.1	67.5	3458
Crohn's disease	9	65.5	67.3	3204	12	84.7	71.2	3651
Tick-borne encephalitis	6	83.3	69.2	801	6	100.0	96.9	720
Lung cancer	8	72.3	69.1	3972	14	84.2	83.2	5240
Rheumatoid arthritis	13	75.0	63.4	19440	13	85.2	71.7	21240
	$k = 7, \alpha = 1.5$				$k = 10, \alpha = 1.5$			
Autoimmune disorder	16	86.5	72.4	3971	16	86.5	74.6	4025
Crohn's disease	14	84.7	74.0	4044	16	86.1	75.3	4400
Tick-borne encephalitis	6	100.0	96.9	824	6	100.0	96.9	820
Lung cancer	16	85.1	83.9	5354	16	84.4	85.0	5517
Rheumatoid arthritis	13	83.4	75.0	22320	14	85.0	77.8	23060

The results of our model using the 5-fold CV procedure are shown in Table 1. We also used the MDR method [17] with $k = 4, 5$ for comparison, since MDR could not finish for larger k 's. For $k = 5$ a random search option was used for the rheumatoid arthritis data set, due to the large number of SNPs in it.

Table 2. Sensitivity and specificity of the MDR method [17] obtained from the 5-fold CV

Data set	$k=4$		$k=5$	
	Sens(%)	Spec(%)	(Sens)(%)	(Spec)(%)
Autoimmune	50.9	55.3	54.3	60.2
Crohn's	48.6	50.0	49.6	42.5
Tick-borne	52.1	67.2	50.0	67.5
Lung cancer	50.0	57.8	48.8	58.9
Rheumatoid	49.7	54.9	41.1	54.6

Our model achieved the best sensitivity and specificity when using $k = 10$. Even for $k = 5$, our proposed model outperformed MDR significantly. The percentages of sensitivity (resp. specificity) increase reached 70.8 (67.1), 54.9 (12.1), 100.0 (42.5), 107.3 (31.3), and 72.5

(41.3) for Crohn's disease, autoimmune disorder, tick-borne encephalitis virus, rheumatoid arthritis, and lung cancer, respectively. Note also that except for tick-borne encephalitis (which has very few cases), the numbers of clusters formed for the other four diseases are very similar, even though the number of cases across these disease data sets vary from 144 to 460. This seems to indicate that natural clusters that are independent of the number of cases are being formed.

Thus assuming that the cases in our data sets are representative of the general case population, the number of clusters for any of these diseases would probably remain unchanged even if the number of cases increases significantly.

Table 3. Sensitivity, specificity and runtimes obtained for the DNF method and the CPS method [4] for different data sets. The DNF results are from the 5-fold cross-validation procedure, and the CPS results are reported from [4], which were obtained from a leave-out cross-validation procedure.

Data set	DNF			CPS		
	Sens (%)	Spec (%)	runtime (sec)	Sens (%)	Spec (%)	Runtime (hour)
Crohn's	86.1	75.3	4400	80.0	89.9	1189
Tick-borne	100.0	96.9	820	80.2	92.4	6.3
Autoimmune	86.5	74.6	4025	79.0	89.1	17400

We also compared the results of our method and with those of the CPS method. Our method increased sensitivity by 7.6%, 7.7%, 26.6% for Crohn's disease, autoimmune disorder, and tick-borne encephalitis virus, respectively. However, the specificity was decrease by 16.3% and 16.3% for the first

two data sets, but increased by 4.9% for the last data set. Our methods are significantly faster than the CPS. For example, it only took our method 820 seconds for tick-borne data, but it took the CPS method 6.3 hours, even though we used a somewhat slower CPU than that used by CPS (a 1.8 GHz Pentium M vs. a 3.2 Ghz Pentium 4).

4 Conclusion

We proposed 0/1 IP problems to identify k -SNP marker sets that predict an individual's disease status from their genotype data. Our method demonstrated significant improvement in performance compared with several existing methods. Using the novel DNF technique our model can simultaneously detect multiple k -SNP marker sets which correspond to different genetic subgroups from data of large scale case-control studies, and do so very time-efficiently. With further refinement of the model, our method has promise for the analysis of genome-wide association study data.

Acknowledgments. We would like to thank Dr. Dumitru Brinza for providing the data sets used in this work.

References

1. Ahuja, R.K.K., Magnanti, T.L., Orlin, J.B.: Network Flows: Theory, Algorithms, and Applications. Pearson Education, London (1993)
2. Atamurk, A., Savelsbergh, M.: Integer Programming Software Systems. Annals of Operations Research 140(1), 67–124 (2005)
3. Brinza, D., Zelikovsky, A.: Combinatorial Analysis of Disease Association and Susceptibility for Rheumatoid Arthritis SNP Data. In: Proc. of 15th Genetic Analysis Workshop (GAW15), pp. 6–11 (2006)

4. Brinza, D., Zelikovsky, A.: Combinatorial methods for disease association search and susceptibility prediction. In: Bücher, P., Moret, B.M.E. (eds.) WABI 2006. LNCS (LNBI), vol. 4175, pp. 286–297. Springer, Heidelberg (2006)
5. Brinza, D., Zelikovsky, A.: Design and Validation of Methods Searching for Risk Factors in Genotype Case-Control Studies. *Journal of Computational Biology* 15, 81–90 (2008)
6. Daly, M.J., Rioux, J.D., Schaffner, S.F., Hudson, T.J., Lander, E.S.: High-resolution haplotype structure in the human genome. *Nat. Genet.* 29, 229–232 (2001)
7. Dutt, S., Ren, H.: Discretized Network Flow Techniques for Timing and Wire-Length Driven Incremental Placement with White-Space Satisfaction. *IEEE Trans. of VLSI* (accepted for publication, 2008)
8. Dutt, S., Ren, H., Suthar, V.: A Network-Flow Approach to Timing-Driven Incremental Placement for ASICs. In: *Proc. IEEE Int'l Conf. CAD (ICCAD)*, pp. 375–382 (2006)
9. Hirschhorn, J.N., Daly, M.J.: Genome-wide association studies for common diseases and complex traits. *Nature Rev. Genet.* 6, 95 (2005)
10. Nahapetyan, A., Pardalos, P.: Adaptive Dynamic Cost Updating Procedure for Solving Fixed Charge Network Flow Problems. *Computational Optimization and Appl. Jour.* 39(1), 37–50 (2008)
11. Li, J.: A novel strategy for detecting multiple loci in Genome-Wide Association Studies of complex diseases. *International Journal of Bioinformatics Research and Applications* 4, 150–163 (2008)
12. Li, J.: Prioritize and Select SNPs for Association Studies with Multi-Stage Designs. *Journal of Computational Biology* 15, 241–257 (2008)
13. McCarthy, M.I., et al.: Genome-wide association studies for complex traits: consensus, uncertainty and challenges. *Nat. Rev. Genet.* 9, 356–369 (2008)
14. Musani, S.K., et al.: Detection of Gene Gene Interactions in Genome-Wide Association Studies of Human Population Data. *Hum Hered* 63, 67–84 (2007)
15. Nelson, M., Kardia, S., Ferrell, R.: A combinatorial partitioning method to identify multi-locus genotypic partitions that predict the quantitative trait variation. *Genome Res.* 11, 2115 (2001)
16. Ren, H., Dutt, S.: Algorithms for Simultaneous Consideration of Multiple Physical Synthesis Transforms for Timing Closure. In: *Proc. IEEE Int'l Conf. CAD (ICCAD)* (accepted for publication) (2008)
17. Ritchie, M.D., Hahn, L.W., Moore, J.H.: Software for MDR and MDR Permutation Testing module Power of multifactor dimensionality reduction for detecting gene-gene interactions in the presence of genotyping error, missing data, phenocopy, and genetic heterogeneity. *Genet Epidemiol.* 24, 150–157 (2003)
18. Spinola, M., et al.: Association of the PDCD5 Locus With Lung Cancer Risk and Prognosis in Smokers. *J. Clin. Oncol.* 24, 1672–1678 (2006)
19. Ueda, H., et al.: Association of the T-cell regulatory gene CTLA4 with susceptibility to autoimmune disease. *Nature* 423, 506–511 (2003)
20. Zhang, Y., Liu, J.S.: Bayesian inference of epistatic interactions in case-control studies. *Nat. Genet.* 39, 1167 (2007)

Biclustering Expression Data Based on Expanding Localized Substructures

Cesim Erten¹ and Melih Sözdinler²

¹ Computer Engineering,
Kadir Has University,
Istanbul 34083 Turkey

² Computer Science and Engineering,
Işık University,
Sile 34980 Turkey

Abstract. Biclustering gene expression data is the problem of extracting submatrices of genes and conditions exhibiting significant correlation across both the rows and the columns of a data matrix of expression values. We provide a method, LEB (Localize-and-Extract Biclusters) which reduces the search space into local neighborhoods within the matrix by first localizing correlated structures. The localization procedure takes its roots from effective use of graph-theoretical methods applied to problems exhibiting a similar structure to that of biclustering. Once interesting structures are localized the search space reduces to small neighborhoods and the biclusters are extracted from these localities. We evaluate the effectiveness of our method with extensive experiments both using artificial and real datasets.

1 Introduction

Clustering refers to the process of organizing a set of input vectors into clusters based on similarity specified according to some predefined distance measure. In many cases it is more desirable to simultaneously cluster the dimensions as well as the vectors themselves. This special instance of clustering, referred to as *biclustering*, was introduced by Hartigan [14]. In addition to the areas such as data mining and pattern recognition, biclustering has found many applications in bioinformatics, specifically in microarray analysis, drug activity analysis, and motif detection [5, 23, 19, 16, 7, 2, 4, 21]. It is different from the existing traditional clustering paradigm. Although traditional one-way clustering provides valuable information with regards to a global perspective, extracting local substructures via biclustering may help build intuition on both dimensions of the data.

In gene expression analysis, data is assumed to be arranged in a matrix, where each gene corresponds to a row and each condition to a column. After regrouping of rows and columns of the matrix, a bicluster can then be defined as a submatrix satisfying certain criteria. These criteria maybe used to categorize biclusters into certain classes. In a *constant-valued* bicluster, values in the submatrix corresponding to the bicluster are all equal. A bicluster with *constant-rows* on the

other hand is defined as a submatrix consisting of values obtained by $\delta \odot \alpha_i$, where δ is the typical value within the submatrix, α_i is the adjustment for the i^{th} row of the submatrix. Depending on the choice of additive or multiplicative adjustment model, \odot corresponds to addition or multiplication respectively. *Constant-column* biclusters are defined symmetrically. Finally *coherent-valued* biclusters are those submatrices with values defined as $\delta \odot \alpha_i \odot \beta_j$, where α_i, β_j are the adjustments for the i^{th} row and the j^{th} column of the submatrix respectively. Again additive or multiplicative adjustment maybe applied by choosing the \odot operator appropriately. See the recent survey of Madeira et al. for further details [17]. We note that the above definitions for biclusters apply to “ideal” conditions where no noise is present in data. The variety of techniques applied to handle noisy data creates the main distinction between the previous approaches.

2 Previous Work

One of the early approaches for biclustering expression data is that of Cheng and Church [12]. They provide a greedy, iterative algorithm with running time $O(mn)$, where m and n are the dimensions of the data matrix. The *mean squared residue* score is defined and the algorithm greedily inserts/removes rows and columns to arrive at a certain number of biclusters achieving some predefined score value. Order-Preserving Sub Matrix(OPSM) [5] is another greedy, iterative algorithm, that finds a statistically significant bicluster at each iteration. It greedily runs over the data matrix and enlarges the best bicluster and continues until there is no reported bicluster. The time complexity of OPSM is $O(nm^3l)$ where n and m are dimensions of input matrix and l is the number of output biclusters. The algorithm does not scale well for high dimensional data. Conserved gene expression motifs or xMOTIFs [19] is another greedy algorithm which finds all biclusters at a single run. ISA [7] introduces a statistical approach to the biclustering problem. It requires normalized data with mean 0, variance 1 and assigns weights according to z-scores that represents significant behavior with similar weights. It has been applied to determine cis-regulatory modules in the yeast dataset. Coupled two way clustering is proposed by Getz et al. [13]. First stable forms on submatrices that are used to divide the original dataset are found. Then one-way clustering is applied recursively on a single dimension of the submatrices until there is no newly found stable submatrix. They guarantee that each submatrix pair is not encountered more than once. Their success depends on the performance of one dimensional clustering algorithms such as K-means, Hierarchical, SOM.

Several graph-theoretical approaches have also been suggested. Prelic et al. provided a divide-and-conquer algorithm, Bimax [21], that runs on discretized binary data. Since they rely on a discretization strategy, the results of Bimax may vary according to the employed strategy. Besides binary data makes it harder to focus on coherent values and thus applies only to constant biclusters. In SAMBA [23, 22] the data matrix is viewed as a bipartite graph where the genes/conditions constitute the layers of the bipartite graph and edges in the

graph correspond to the expression changes. The goal is to find out heavy bicliques inside the graph. Running time of SAMBA is exponential on the size of the conditions set in a maximum bounded biclique because of the employed exhaustive bicluster enumeration. A similar model is constructed in [2] where crossing minimization in unit-weight bipartite graphs is used as a means to extract bicliques corresponding to biclusters in the data matrix.

3 LEB- Localize and Extract Biclusters

Given an input data matrix M , the idea behind our algorithm is to first group together rows and columns of M in such a way that correlated rows/columns are “localized”, that is rows/columns exhibiting similar patterns are placed in nearby locations within M . Once localization is complete our search space should be limited to small neighborhoods within the matrix. Starting with small local submatrices we evaluate the score of a submatrix corresponding to a possible bicluster and grow it if necessary until we arrive at a bicluster with a fair score.

3.1 Graph-Theoretical Preliminaries

The vertex set in a bipartite graph is partitioned into two so that no edge exists between a pair of vertices within the same partition. The data matrix M can be turned into a weighted bipartite graph G_M . The vertices in one partition of G_M correspond to the rows and the vertices in the other partition correspond to the columns. Each entry $a_{i,j} \in M$ is assigned to the weight of the edge (i, j) in G_M . The maximum edge biclique problem in unweighted bipartite graphs is NP-hard [20]. Given the above correspondence this implies finding the maximum size constant-valued bicluster is also NP-hard since a special case of this can be used to solve the biclique problem. If the data matrix consists only of 0, 1 values then extracting all interesting constant-valued biclusters reduces to the problem of maximal biclique generation problem. This problem is also NP-hard and various versions of it have been studied previously in [3]. The Bimax algorithm makes use of the introduced techniques [21].

A well-studied graph theoretical problem is that of crossing minimization in bipartite graphs. Given a bipartite graph the goal is to arrange the vertices in such an order that minimizes the total number of edge crossings, if the partitions were to be on two parallel lines and each edge drawn using a straight-line segment. This problem is also NP-hard even when one of the partitions is fixed. The connection between the biclique generation and the bicluster extraction problems, and the observation that minimizing crossings requires grouping heavy bicliques “locally” together is the main motivation behind our method. We note that similar approaches have been taken [2]. The first step in such approaches is to discretize the data so as to achieve a unit-weight bipartite graph which may then be rearranged to provide few edge crossings. However the discretization of data may cause loss of valuable data. The problem could be remedied by considering the data matrix in its original form and applying a crossing minimization

on a weighted bipartite graph. However choosing an appropriate crossing minimization heuristic and applying it in a manner suitable for the biclustering problem is an important task. It has been shown recently in [10] that heuristics that are simple generalizations of well-known heuristics and approximations of the unweighted settings do not work well under the more general weighted settings of the problem. In LEB we apply a similar idea of “localization” that makes use of crossing minimization in bipartite graphs. However we do not have a discretization/normalization step to convert the weighted bipartite graph into an unweighted one as this might cause data loss and produce erroneous output. Instead we apply crossing minimization directly on the original weighted graph. An important property of our method is that we use a 3-approximation algorithm which has been shown to work well in practice as well, for the general weighted version of the crossing minimization problem. Additionally our localization procedure incorporates this heuristic in a manner that successfully handles the discrepancy caused by noise in the data.

3.2 Bicluster Localization

A pseudocode of this first phase is provided in Algorithm 1. The localization proceeds in two steps. The first step is to provide a good initial localization. This is achieved by reordering the rows and columns in a way that the resulting bipartite graph corresponding to M contains few (weighted) crossings between the edges. To this aim first the columns are fixed and the rows are reordered using a 3-approximation algorithm for minimizing crossings in weighted bipartite graphs. The result below follows:

Theorem 1 (Cakiroglu et al. [10]). *Fixing the columns of M , Phase-1.1 orders the rows in such a way that the weighted crossings in the resulting bipartite graph is at most three times the optimum.*

Next the rows are fixed and the columns are reordered using the same method. This procedure of fixing one dimension and reordering the other is continued until it converges, that is no further improvement can be made or a certain iteration count is reached. We have verified that if the input data is noise-free then this initial localization step is usually enough to identify bicliques and extract the biclusters. However with real data finding biclusters is not an easy task because of large amounts of noise. In order to deal with noise, the next step in localization is *Adaptive Noise Hiding*. The weighted edges in the graph that correspond to noise in the original input data are removed in this step. Sliding a window around the perimeter of each entry in the data matrix, where $(r \pm 1, c)$, $(r, c \pm 1)$, $(r \pm 1, c \pm 1)$ constitute the perimeter of the entry at (r, c) , we check whether the window satisfies a threshold density in terms of the number of nonzero weighted edges. If it does not, the entries on the perimeter are considered *suspicious*. Once sliding is finished we find the *most suspicious* weight and remove all suspicious entries with the corresponding weight. We adaptively apply our crossing minimization based *Initial Localization* step on the new matrix and continue noise hiding after

Algorithm 1. Phase-1: Bicluster Localization

```

/* Phase-1.1: Initial Localization */
repeat
  /* Fix columns, order rows to minimize crossings */
  for all  $r \in R$  do
    leftsum = 0; rightsum =  $\sum_{i=2}^{|C|} a_{r,i}$ ;
    for all  $c \in C$  do
      if leftsum  $\geq$  rightsum then break;
      leftsum+ =  $a_{r,c}$ ; rightsum- =  $a_{r,c+1}$ ;
    end for
     $P_c = P_c \cup \{u\}$ ;
  end for
  for all  $c \in C$  do
     $\sum_{i=1}^c a_{v,i} \times \sum_{i=c}^{|C|} a_{u,i} \leq \sum_{i=1}^c a_{u,i} \times \sum_{i=c}^{|C|} a_{v,i}$ 
    defines a total order  $u \preceq v$ , Sort  $P_c$  using  $\preceq$ 
  end for
  /* Fix rows, order columns to minimize crossings */
  Switch  $R$  and  $C$ , Repeat above procedure.
until no change in row/column ordering or enough iterations

/* Phase-1.2: Adaptive Noise Hiding */
repeat
  for all pairs  $(r, c)$  where  $r \in R, c \in C$  do
    Neighboring Pairs  $NP = \{a_{r\pm 1,c}, a_{r,c\pm 1}, a_{r\pm 1,c\pm 1}\}$ 
    Let count be the number of elements in  $NP$  with nonzero weight
    if count  $\leq$  threshold then Store those entries as suspicious;
  end for
  Find the most suspicious weight,  $MS$ 
  Hide entries  $a_{r,c}$  with weight equal to  $MS$  and Run Phase-1.1
  if no suspicious entries then threshold + +
until threshold reaches max-threshold-density

```

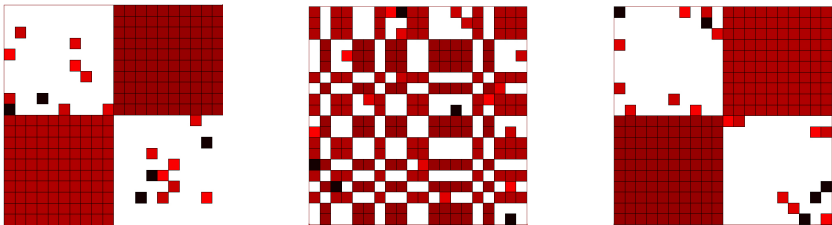


Fig. 1. Assumed noise is 0.05. (a)Initial artificial design with 2 biclusters of $K_{10,10}$; (b)After Phase-1.1: Initial Localization; (c)After Phase-1.2: Adaptive Noise Hiding

incrementing the threshold density. The removal of the *suspicious* edges and the crossings in the corresponding weighted bipartite graph couple each other in terms of noise removal. Each time the rows/columns are reordered to reduce the number of crossings, new suspicious pairs are created to be removed in the next iteration of the procedure. We note that our application of the crossing minimization is two-folds. Besides providing a good initial localization, crossing minimization is also used to handle noise removal. A simple run showcase is provided in Figure [11](#).

3.3 Bicluster Extraction

The next step in LEB is to extract biclusters from the reorganized data matrix by considering local neighborhoods. We introduce two parameters α , the size of a neighborhood and γ , the required score of a bicluster.

We divide the data matrix into bags of size $\alpha \times \alpha$. We do a traversal over the bag set and try to enlarge the bags. All bags are unmarked initially. At each iteration during the traversal we first mark a bag b_i . We compute the *evaluation score* of b_i with the next unmarked bag b_j in the x -direction and construct their union if the score “satisfies” the γ constraint. We mark b_j and continue with the union as the current bag. Once we check over all bags in the x -direction, we continue with the y -direction and follow the same procedure. Thus at the end of one such iteration we have the boundaries of a bicluster determined. We continue the traversal starting from an unmarked bag and follow the same enlargement procedure. Once all bags are marked we are left with nonoverlapping, different sized bags corresponding to biclusters all of which satisfy the γ score.

For constant biclusters the *evaluation score* of b_i, b_j is the ratio of the number of most frequent weight in the union to the size of the union. A score satisfies the γ constraint if it is larger than γ . We note that we have an additional initial traversal step for constant biclusters where we remove bag b_i that scores lower than γ when evaluated with the empty set.

On the other hand for coherent biclusters we first define the H -value of a submatrix [\[12\]](#). Assume the submatrix consists of I rows and J columns. The residue R of an entry (i, j) is

$$R_{I,J}(i, j) = a_{i,j} - a_{Ij} - a_{iJ} + a_{IJ} \tag{1}$$

where a_{iJ} is the mean of row i , a_{Ij} is the mean of column j and a_{IJ} is the mean of the submatrix. H-value is defined as,

$$H_{I,J}(i, j) = \frac{1}{|I||J|} \sum_{i=0, j=0} (RS_{I,J}(i, j)^2) \tag{2}$$

The *evaluation score* of two bags is the difference between their H -values. A score satisfies the γ constraint if it is less than γ in this case.

Table 1. Experiments on *Arabidopsis thaliana* dataset

Algorithm	Maximum Bicluster	Minimum Bicluster	Dimension1 Average	Dimension2 Average	Bicluster Count	H-values
LEB	630x66	3x6	60,13043	6,945652	92	398,009
BIMAX	10x4	4x4	4.428301	4.065273	50189	254,3467
CC	690x69	242x69	531.4200	69.00000	100	1654,066
ISA	234x5	198x2	220.3250	2.537500	80	1374,401
OPSM	2x57	10x5	10.33333	15.41667	12	673,0505

3.4 Running Time of LEB

With a straightforward implementation the running time of the *Initial Localization* step in the first phase of LEB is $O(|R| \times |C| + |R| \log |R| + |C| \log |C|)$, where $|R|, |C|$ denote the sizes of the rows, columns of the data matrix respectively. The *repeat* loop of this step is iterated until the minimization of crossings converges or a constant number of iterations is achieved. We note that in all the experiments conducted, the method converges after a few iterations. Similar reasoning applies to the *Adaptive Noise Hiding* step. The *Bicluster Extraction* phase requires time $O(|R| \times |C|)$ as the size of a neighborhood (bag) is constant and each bag is marked once throughout the extraction. Therefore the running time of the first phase of LEB is almost linear in terms of the input size.

4 Experimental Results

We used the LEDA C++ Library [18] to implement our algorithm. The codes for the implementation and the experimentation are freely accessible at [1]. In order to test the rest of the algorithms, we used Biclustering Analysis Toolbox (BicAT) [4] and Click and Expander [22]. We chose 5 different algorithms for comparison with LEB. These are the BIMAX [21], CC [12], ISA [7], SAMBA [23, 22], and OPSM [5] algorithms. We experimented on two real datasets, *Arabidopsis thaliana* and the *Yeast Cell Cycle* (*Saccharomyces cerevisiae*). For the thaliana dataset, we evaluate the bicluster sizes and the H-values. For the yeast dataset, we evaluate the enrichment ratios achieved by each of the selected algorithms. We also provide the evaluation results based on the Protein-Protein Interactions similar to those of Prelic et al. [21].

4.1 Experiments on Arabidopsis Thaliana Dataset

This is a widely used plant for bioinformatics applications. It is the first plant that is sequenced with its whole genome. The gene expression dataset used in the context of this experiment is publicly available at <http://arabidopsis.info/>. The maintained database consists of 734 genes and 69 conditions. We have conducted several experiments on this database; see Table 1. All the algorithms have certain execution parameters. For each algorithm under discussion we assign the parameters to the values set in the original descriptions of the mentioned algorithms.

Table 2. *Yeast* dataset experiment

Warfield for OPSM, CC, LEB on <i>Yeast 2884x17</i> dataset for each category						
Functional Category	OPSM		CC		LEB	
	ORF in Bicluster	Func. Enrich.	ORF in Bicluster	Func. Enrich.	ORF in Bicluster	Func. Enrich.
E - Energy Production	543	0,03	55	0,04	100	0,04
G - Amino Acid Metabolism	1282	0,03	51	0,04	186	0,05
M - Other Metabolism	62	0,11	59	0,14	79	0,22
P - Translation	2342	0,03	57	0,19	79	0,09
T - Transcription	1282	0,06	55	0,19	143	0,08
B - Transcriptional control	124	0,08	42	0,08	152	0,08
F - Protein Fate	2342	0,06	51	0,14	83	0,08
O - Cellular Org.	2342	0,04	51	0,10	46	0,11
A - Transport and Sensing	124	0,13	59	0,07	143	0,10
R - Stress and Defense	62	0,06	42	0,05	105	0,06
D - Genome Maintenance	62	0,11	51	0,10	293	0,14
C - Cellular Fate / Org.	196	0,47	51	0,47	111	0,48
U - Uncharacterized	124	0,06	51	0,08	79	0,13

The settings are as follows: For LEB ($\gamma = 10, \alpha = 2$), for BIMAX ($D_{scrz} = 0.6, \alpha = 4, \beta = 4$), for CC ($\rho = 13, \delta = 100, \alpha = 1.2, outputBiclusters = 100$), for ISA ($\rho = 13, t_g = 2.0, t_c = 1.0, n = 500$), and finally for OPSM ($\gamma = 100$).

In terms of the number of output biclusters CC and ISA perform well. However one problem with these algorithms is the size of biclusters. For instance CC provides a bicluster that is almost the size of input data; the conditions set is represented completely in all the resulting biclusters. ISA has a similar problem with the conditions set, since the biclusters underrepresent the conditions providing little clue as to correlation among them. To further examine the BIMAX algorithm we experiment with various parameter values. We increase the discretization value to 0.6 and we increase α, β values. In that setting BIMAX finds many biclusters but there is no biclusters larger than $10x4$. On the other hand, LEB provides biclusters of reasonable sizes. The resulting H-values are evaluated as a performance measure. LEB provides the second lowest H-value scores; it is likely that highly correlated biclusters are extracted with LEB.

We note that BIMAX has a problem with low discretization parameter values. For instance, when the discretization value is set to 0.2, it extracts millions of biclusters which creates a difficulty in analyzing such an output. Another severe problem is that of execution time. For specific parameter values it takes more than several days to execute. OPSM does not provide better results even if we try several settings for the parameter values. It only extracts 12 biclusters in *Arabidopsis thaliana* dataset, although the size of the dataset is considerably large. We made observations regarding parameter values of LEB. Assigning small values to α , such as 3, provides better results in extracting reasonable size biclusters with low H-value scores.

Table 3. *Yeast* dataset experiment with different parameter settings for LEB. The abbreviations used for the categories are the same as in Table 2

Yeast Results for OPSM, CC, LEB1, LEB2, LEB3													
Alg. Name	E	M	G	P	T	B	F	O	A	R	D	C	U
LEB1	0.04	0.05	0.19	0.09	0.09	0.08	0.08	0.11	0.10	0.05	0.14	0.48	0.13
LEB2	0.07	0.05	0.16	0.12	0.14	0.07	0.11	0.10	0.10	0.06	0.17	0.45	0.10
LEB3	0.11	0.05	0.16	0.16	0.11	0.08	0.13	0.10	0.10	0.07	0.19	0.51	0.10
CC	0.04	0.04	0.14	0.21	0.24	0.1	0.14	0.10	0.07	0.06	0.10	0.47	0.08
OPSM	0.03	0.03	0.11	0.03	0.06	0.08	0.06	0.04	0.13	0.06	0.11	0.47	0.06

4.2 Experiments on Yeast Dataset

Our second set of experiments are conducted on the Yeast Cell Cycle (*S.cerevisiae*) dataset [12]. Most of the previously suggested biclustering methods have been tested for this dataset. It has 2884 genes and 17 conditions. Since *Yeast* dataset was categorized in terms of the functionality of each of the genes at MIPS [15], we are able to test the *enrichment ratio* of each bicluster. The identification of the categories of genes is done similar to [9]. There are 13 pre-identified categories. Given a bicluster the ratio of the number of genes specified in a category to the number of genes in the bicluster provides a possible enrichment. For a specific category we choose the highest enrichment value among all biclusters as the enrichment ratio of the category. This is a ratio between 0 and 1, see Table 2. For this experiment we do not evaluate small biclusters that contain less than 40 genes. OPSM in general fails to enrich biclusters. LEB performs better in 7 categories, CC in 3, and OPSM in only 1. CC and LEB have a draw in one category and all algorithms have a draw in another category. Settings of CC and OPSM are the original default parameters provided in the papers describing the methods [12, 5]. For LEB, $\gamma = 100$, $\alpha = 4$ are the parameter settings. Table 3 provides the statistics of enrichment values for each category where we append three different runs of LEB (LEB1($\gamma = 100$, $\alpha = 4$), LEB2($\gamma = 50$, $\alpha = 3$), LEB3($\gamma = 25$, $\alpha = 3$)) with different parameter settings. LEB performs better than CC and OPSM for various different parameter settings as well.

Next we use the FuncAssociate tool [8] in order to evaluate the algorithms with regards to GO accepted categories. FuncAssociate computes the hypergeometric functional score by using *Fisher's Exact Test*. For each significance level, we provide the enrichment ratio of the biclusters; see Figure 2. We note that we do not run BIMAX and ISA for this dataset. ISA does not provide any biclusters although many parameter settings were tried. BIMAX has a problem with the number of biclusters. There is a significant number of biclusters but many of them are duplicates. Furthermore parameter selection is another problematic issue. However to gain a better insight on a comparative evaluation, in the figure we provide the results of these two algorithms using reported results from [9, 21, 11]. The values in consideration are the adjusted p-values gathered from the FuncAssociate tool [8]. According to the adjusted p-values, LEB has an increasing enrichment ratio when significance level increases as expected. It is good at covering genes with large amounts with high hit ratio inside the bicluster.

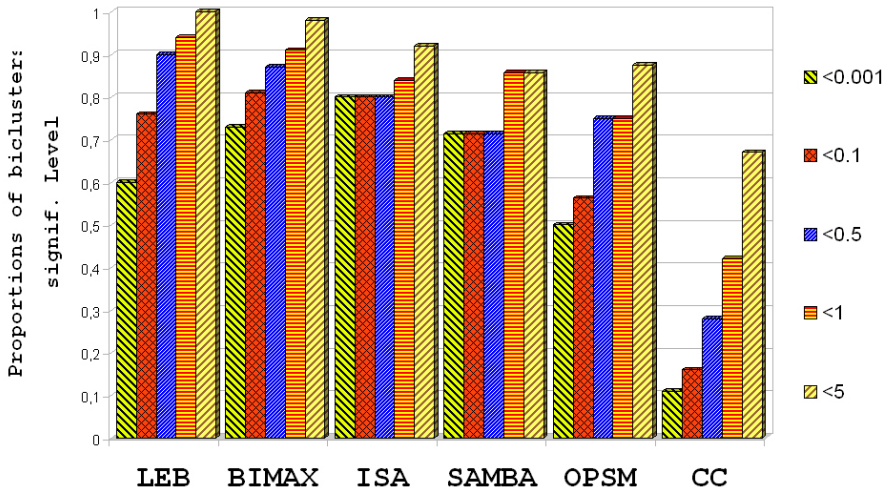


Fig. 2. Proportion of biclusters significantly enriched by any GO biological category of Yeast (*S.cerevisiae*) for LEB ($\gamma = 10, \alpha = 2$), BIMAX, ISA, SAMBA, OPSM, and CC. Note that the results of BIMAX and ISA are those reported in [9, 21, 11].

Table 4. Yeast PPI experiment, Averages of all hit ratios of biclusters

PPI Experiment for LEB, OPSM, CC, BIMAX and ISA on <i>Yeast 2884x17</i> dataset					
Algorithm Name	LEB	OPSM	CC	BIMAX	ISA
Hit Ratio	0,05783	0,03004	0,04968	0,04458	0

Among all tested results of LEB, CC, SAMBA and OPSM, LEB provides better results than the other algorithms. Including the reported results [9, 21, 11] of BIMAX and ISA, LEB still performs better in most of the categories.

Additionally, as a final experiment, we have done the Protein-Protein Interactions (PPI) experiments where we use the data from [6]. According to this data, there are relations which are categorized as *physical* or *complex* among each pair of genes within the Yeast dataset. If there is no edge for a given gene pair in the PPI graph, we say that there is neither a physical nor a complex relation. In our experiment, for each bicluster of the Yeast Cycle dataset, we extract the genes and among these genes we define artificial relations as edges. Each gene is connected to all other genes. As the next step, for randomly selected artificial relations, we check to see whether there are any real *complex* or *physical* relations between source gene and target gene pairs at the PPI Network. If there is a relation, we increment the number of hits, h . We also store the number of queries as n . As a result, for the bicluster under consideration, we compute a hit ratio defined as h/n . High hit ratio means that genes in a bicluster are more correlated in terms of protein-protein interactions. Table 4 provides the related ratios as the averages of hit ratios among all generated biclusters. According to these results, LEB has the best hit ratio among others. In this experiment, ISA

has a hit ratio of 0 since no output biclusters are generated for the suggested parameters of the algorithm.

5 Conclusion

We proposed an algorithm LEB (Localize and Extract Biclusters) for biclustering gene expression data. The main idea behind the method is the observation that minimizing crossings in weighted bipartite graphs provides a partitioning that gathers probable biclusters within close neighborhoods. The algorithm is simple to implement and adopt. The issue of suitable parameter setting is not a problem as compared to most of the other alternative methods. Experiments on both real data and artificially constructed data indicate that LEB performs fairly well in general. One direction that still needs to be examined is to establish a formal theoretical connection between the weighted biclique problem and various more general definitions of biclustering. Embedding other weighted biclique heuristics within the framework of LEB and testing them against the current LEB, with the underlying crossing minimization operation would be a valuable comparison.

References

1. Leb, <http://hacivat.khas.edu.tr/~cesim/lebsource.rar>
2. Abdullah, A., Hussain, A.: A new biclustering technique based on crossing minimization. *Neurocomputing* 69(16-18), 1882–1896 (2006)
3. Alexe, G., Alexe, S., Crama, Y., Foldes, S., Hammer, P.L., Simeone, B.: Consensus algorithms for the generation of all maximal bicliques. *Discrete Appl. Math.* 145(1), 11–21 (2004)
4. Barkow, S., Bleuler, S., Prelic, A., Zimmermann, P., Zitzler, E.: Bicat: a biclustering analysis toolbox. *Bioinformatics (Oxford, England)* 22(10), 1282–1283 (2006)
5. Ben-Dor, A., Chor, B., Karp, R., Yakhini, Z.: Discovering local structure in gene expression data: the order-preserving submatrix problem. In: *RECOMB 2002: Proceedings of the sixth annual international conference on Computational biology*, pp. 49–57. ACM, New York (2002)
6. Ben-Hur, A., Noble, W.S.: Kernel methods for predicting protein–protein interactions. *Bioinformatics* 21(1), 38–46 (2005)
7. Bergmann, S., Ihmels, J., Barkai, N.: Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical review. E, Statistical, nonlinear, and soft matter physics* 67(3 Pt 1) (March 2003)
8. Berriz, G.F., King, O.D., Bryant, B., Sander, C., Roth, F.P.: Characterizing gene sets with funcassociate. *Bioinformatics* 19(18), 2502–2504 (2003)
9. Bryan, K., Cunningham, P.: Bottom-up biclustering of expression data. In: *Proceedings of the 2006 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB 2006*, vol. (4133177), pp. 232–239 (2006)
10. Çakiroglu, O.A., Erten, C., Karatas, Ö., Sözdinler, M.: Crossing minimization in weighted bipartite graphs. *Journal of Discrete Algorithms* (2008), doi:10.1016/j.jda.2008.08.003

11. Chen, K., Hu, Y.-J.: Bicluster analysis of genome-wide gene expression. In: 2006 IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology, 2006. CIBCB 2006, pp. 1–7 (September 2006)
12. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Altman, R., Bailey, T.L., Bourne, P., Gribskov, M., Lengauer, T., Shindyalov, I.N. (eds.) Proceedings of the 8th International Conference on Intelligent Systems for Molecular (ISMB 2000), Menlo Park, CA, August 16–23, pp. 93–103. AAAI Press, Menlo Park (2000)
13. Getz, G., Levine, E., Domany, E.: Coupled two-way clustering analysis of gene microarray data. In: Proc. Natl. Acad. Sci. USA, pp. 12079–12084 (2000)
14. Hartigan, J.A.: Direct clustering of a data matrix. *Journal of the American Statistical Association* 67(337), 123–129 (1972)
15. Mewes, H.W., Frishman, D., Güldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Münsterkötter, M., Rudd, S., Weil, B.: Mips: a database for genomes and protein sequences. *Nucleic Acids Res.* 30(1), 31–34 (2002)
16. Kluger, Y., Basri, R., Chang, J.T., Gerstein, M.: Spectral biclustering of microarray data: coclustering genes and conditions. *Journal Genome Res* PMID 12671006 13, 703–716 (2003)
17. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. on Comp. Biol. and Bioinformatics (TCBB)* 1(1), 24–45 (2004)
18. Mehlhorn, K., Naher, S.: *Leda: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, Cambridge (1999)
19. Murali, T.M., Kasif, S.: Extracting conserved gene expression motifs from gene expression data. In: Pacific Symposium on Biocomputing, pp. 77–88 (2003)
20. Peeters, R.: The maximum edge biclique problem is np-complete. *Discrete Appl. Math.* 131(3), 651–654 (2003)
21. Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., Buhlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics* 22, 1122–1129 (2006)
22. Sharan, R., Maron-katz, A., Shamir, R.: Click and expander: A system for clustering and visualizing gene expression data. *Bioinformatics* 19, 1787–1799 (2003)
23. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. *Bioinformatics* 18(suppl. 1) (2002)

Constrained Fisher Scores Derived from Interaction Profile Hidden Markov Models Improve Protein to Protein Interaction Prediction

Alvaro J. González and Li Liao*

University of Delaware, Computer Science Department,
421 Smith Hall, Newark DE 19716, USA
{alvaro, lliao}@cis.udel.edu

Abstract. Protein-protein interaction plays critical roles in cellular functions. In this paper, we propose a computational method to predict protein-protein interaction by using support vector machines and the constrained Fisher scores derived from interaction profile hidden Markov models (ipHMM) that characterize domains involved in the interaction. The constrained Fisher scores are obtained as the gradient, with respect to the model parameters, of the posterior probability for the protein to be aligned with the ipHMM as conditioned on a specified path through the model state space, in this case we used the most probable path –as determined by the Viterbi algorithm. The method is tested by leave-one-out cross validation experiments with a set of interacting protein pairs adopted from the 3DID database. The prediction accuracy measured by ROC score has shown significant improvement as compared to the previous methods.

Keywords: Protein-protein interaction, Hidden Markov models, Support vector machines, Domains, 3DID.

1 Introduction

Recently, prompted by high throughput technologies and the huge amount of data wherein generated, systems biology has become a new powerful paradigm in studying molecular biology and cellular processes. At the center of systems biology is the reverse engineering problem for discovering the underlying biological networks that are responsible for the various phenotypes manifested in cell lines. Predicting protein-protein interaction (PPI) computationally is an important step in that reverse engineering effort.

The computational methods so far developed for PPI prediction have utilized information from various sources at different levels, from primary sequences, to molecular structures, to evolutionary profiles, and with varied performance as reported in the literature (see [5] and references therein). Typically, more sensitive

* Corresponding author.

prediction tends to require extensive information, *e.g.* phylogenetic information, and more specific prediction tends to require more detailed information, *e.g.* the structural information. How to integrate and incorporate various data sources has been a central issue in systems biology, and as we show here it also plays an important role in improving the PPI prediction accuracy.

It is known that protein's 3D structure is essential to its function, for example, the binding sites may require some particular shape to be thermodynamically favorable for the binding. When proteins interact with one another, the interfaces, also called domains, where the interaction happens, also need to take up an appropriate structure, which in turn may impose constraints on the amino acid composition at the interfaces. It has become a common approach for predicting PPI to identify these domains. However, several factors can compromise the efforts of using domain-domain interaction (DDI) to predict PPI.

First, although the domains responsible for binding two proteins together tend to possess certain biochemical properties that dictate some specific composition of amino acids, such compositions are typically not unique enough to be solely relied upon for domain identification –variations are quite common in the multiple sequence alignment of these proteins that contain the same domain. Hidden Markov models are among the most successful efforts to capture the commonalities of a given domain while allowing variations. A collection of hidden Markov models covering many common protein domains and families is available in the Pfam database [7].

More seriously, although corroborated by other evidences such as domain modularity of proteins and shared DDI among PPIs, in most cases experimental verification in support of the DDI-PPI correspondence is still missing [10]. Even when such correspondence is certain, membership of domain families is established at the best, as mentioned above, via probabilistic modeling, therefore false positives are not uncommon. Furthermore, while the interaction sites within domains, as recently demonstrated, play a key role in determining PPI [11], such information is not readily available for many proteins –the dataset of PPIs that have been resolved using crystallography remains relatively small.

In this work, we propose a new computational method to address these issues, in particular by transferring knowledge across sources and at different levels. The method is based on a framework first proposed by Jaakkola *et.al.* [2] [3], which allows for combining generative models and discriminative classifiers. In this case, the generative model is an interacting profile hidden Markov model (ipHMM), recently developed by Fredrich *et.al.* [11], in which the interaction sites within protein domains are taken into account by the model topology. The structural information about the interaction domains is only needed for training the model. Once a model is trained, it can be used to predict interaction sites for proteins with only the sequential information as input. A posterior decoding algorithm yields probabilities for interacting sequence positions and enhances the quality of interaction site predictions.

The discriminative classifier used in this work is a support vector machine (SVM). To leverage the domain information captured in the ipHMM, the SVM

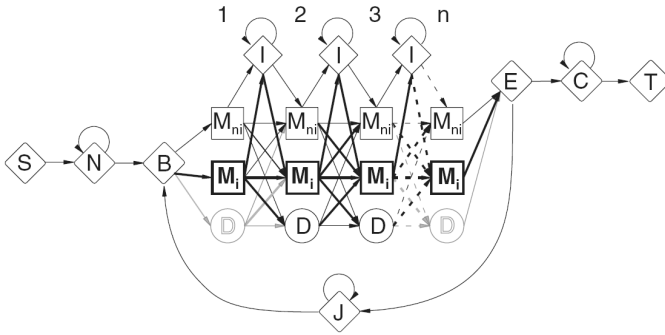


Fig. 1. Topology of the interaction profile hidden Markov model. The match states of the classical pHMM are split into non-interacting (M_{ni}) and interacting (M_i) match states.

is trained on feature vectors that are composed of the constrained Fisher scores, which are defined as the gradient, with respect to the model parameters, of the posterior probability for the protein to be aligned with the ipHMM as conditioned on a specified path through the model state space, in this case we used the most probable path –as determined by the Viterbi algorithm. The method is tested by leave-one-out cross validation experiments with a set of interacting protein pairs adopted from the 3DID database. The prediction accuracy measured by ROC score has shown a significant improvement as compared to the previous methods.

2 Method

2.1 Interaction Profile Hidden Markov Models

Friedrich *et al.* [11] proposed a method for the prediction of interacting sites within protein domains, based on a modified interaction profile hidden Markov model (ipHMM) whose topology takes both structural information and sequence data into account. Every ipHMM is, like profile hidden Markov models (pHMM), a probabilistic representation of a protein domain family. Topology of the ipHMM follows the same restrictions and connectivity of the HMMer architecture, with one exception: that the match states of the classical pHMM are split into a non-interacting (M_{ni}) and an interacting match state (M_i), as shown in Fig. 1. The new kind of states is provided with the same properties of the match state in the classic pHMM architecture, *e.g.* these interacting match states are able to emit all amino acid symbols with certain probabilities, which are the parameters to be fixed according to the training examples.

The parameters of an ipHMM are estimated from a multiple sequence alignment of the member proteins in the domain family, incorporating the annotation on their binding sites –all sequence positions have to be labeled with the corresponding interaction status (0 for non-interacting and 1 for interacting). The model estimation of the ipHMMs is achieved by maximum likelihood. Once

characterized, these ipHMMs have shown to encode an enormous amount of statistical information pertaining to the interaction between the domain and other domains. In this work we will show that this model, although originally created for interacting site prediction, can be turned into a powerful tool for other type of predictions, for instance protein-protein interaction prediction.

2.2 Fisher Scores

Hidden Markov Models (HMM), like the one described in Sect. 2.1, are probability models that can be used to calculate the likelihood of the unannotated sequence being generated by the HMM. Given the structure of the HMM, the model can be fully characterized by finding its emission and transition probabilities following a maximum likelihood approach on the annotated sequences [1]. Once trained, the model can be used to efficiently generate new sequences with high probability of belonging to the model's family, and also to measure the likelihood of an observed sequence to have been generated by the model, which is reason why HMMs are regarded as "generative models". This measure is known as the posterior probability $P(x|\theta)$, where x denotes the sequence, and θ denotes the HMM (or the model's parameters, to be more precise). If there are two models, θ and θ' , and the goal is to rank the affinities of sequence x to θ and to θ' , Bayes rule and the posterior probabilities can be used to calculate $P(\theta|x)$ and $P(\theta'|x)$, and by comparing these two quantities the HMM can be turned into a classifier.

However, when classification is in mind, "discriminative methods" [2], which directly estimate a posterior probability for a class label (as in Gaussian process classifiers) or a discriminant function for the class label (as in support vector machines), have been shown to outperform generative methods. But with discriminative models it is hard to define a principled way to handle missing information and variable length sequences (or vectors), two issues that are taken care of by HMMs, as mentioned before. So clearly, a statistically sound method for combining generative and discriminative models for the ultimate goal of classification is needed. More so for our particular application, in which we desire to discriminate interacting protein pairs from non-interacting ones using support vector machines, but the statistical information about the interactions is contained in an ipHMM (Sect. 2.1). Jaakkola *et.al.* described a method to carry out this combination [2] [3] based on the so called "Fisher score".

The Fisher score is defined as the derivative of the log-likelihood score for the query sequence x with respect to a particular parameter of the model. In this work we will focus on the emission probabilities of the ipHMM, for reasons that will become apparent hereinafter. If the probability of emitting amino acid \tilde{x} from state \tilde{s} is named $\theta_{\tilde{x},\tilde{s}}$, the Fisher score of the model with respect to $\theta_{\tilde{x},\tilde{s}}$ is therefore defined as

$$\frac{\partial}{\partial \theta_{\tilde{x},\tilde{s}}} \log P(x|\theta) = \frac{\varepsilon(\tilde{x},\tilde{s})}{\theta_{\tilde{x},\tilde{s}}} - \varepsilon(\tilde{s}) \quad (1)$$

where $\varepsilon(\tilde{s}) = \sum_{x'} \varepsilon(x', \tilde{s})$ and the summation runs over the 20 different amino acids. The derivation of (1) is detailed in [3]. In this formula, $\varepsilon(\tilde{x}, \tilde{s})$ can be seen as the expected posterior probability of visiting state \tilde{s} and generating residue \tilde{x} from that state. This expected value can be calculated, for any state \tilde{s} and for any emitted amino acid \tilde{x} , from the posterior decoding matrix, which can be found efficiently using the forward and backward algorithms [1] [4]. The literature denotes $\varepsilon(\tilde{x}, \tilde{s})$ and $\varepsilon(\tilde{s})$ as the *sufficient statistics* for the parameter $\theta_{\tilde{x}, \tilde{s}}$ in the model. For this reason we say that the *sufficient statistics* of the entire model are embedded in the Fisher scores.

Using the Fisher scores, any protein sequence can now be characterized in the context of an ipHMM by the Fisher scores of the sequence’s posterior alignment to the model. And many sequences of different length will be mapped to vectors with the same dimensionality in the Fisher vector space. Now it should be clear why the Fisher score provides an elegant way to combine a generative model with a discriminative technique, preserving the advantages of the two approaches. Patel and Liao used this method [5] to successfully predict protein to protein interactions. Their contribution consisted on constructing a dataset from two groups of proteins (say group A and group B), where all the proteins in a group share a common domain, and protein sequences from group A have been observed to interact with protein sequences from group B. Furthermore, the domain shared by the sequences in a group is characterized by an ipHMM. Each protein sequence is aligned to its corresponding ipHMM, and a Fisher vector is calculated by finding the Fisher scores of the alignment with respect to the emission probabilities of a) non-interacting match states and b) interacting match states (these two cases are treated separately). Positive examples are built by concatenating Fisher vectors of interacting sequence pairs, and negative examples are built from non-interacting pairs. Training and testing on this dataset was carried out using support vector machines.

The Fisher scores are calculated with respect to the emission probabilities of match states because it is expected that the statistics of the interaction are implicitly contained in these states (since match states are differentiated according to the interaction in the ipHMM); and remember, the ultimate goal of this work is to predict interacting pairs. Patel and Liao showed this approach yields better prediction results as compared to a baseline model where the positive and negative examples are built using only the log-likelihood scores of the protein alignments to their corresponding ipHMM.

2.3 Constrained Fisher Scores

Although the Fisher score of the log-likelihood score $\log P(x|\theta)$ has been proved to provide a powerful means for predicting interacting protein pairs, it can still be taken one step further. Notice the log-likelihood score is commonly used for the task of family membership prediction. For example, in the context of ipHMMs, evaluation of $\log P(x_0|\theta)$ for a given sequence x_0 gives you the likelihood of protein x_0 to contain the domain characterized by the model, but notice that, in a sense, this measure overlooks the interaction information that is contained in

the model. In Sect. 2.2 we described how this information can be injected into the support vector machine by creating vectors using Fisher scores with respect to emission probabilities of non-interacting/interacting match states.

However, we can prioritize the conservation of the interaction information in the first place, when we define the derivatives that make up the Fisher score. Our focus will now be on the most probable path of hidden states or label s for a given sequence x . Obviously the label sequence s will explicitly differentiate non-interacting from interacting match states. Thus, our quantity of interest is not the likelihood the model assigns to sequences $P(x|\theta)$, but the conditional probability $P(s|x, \theta)$ of the label for a given observation sequence x [6]. Therefore, our new Fisher scores, which we call “constrained” Fisher scores, are defined as:

$$\frac{\partial}{\partial \theta_{\tilde{x}, \tilde{s}}} \log P(s|x, \theta) = \frac{\partial}{\partial \theta_{\tilde{x}, \tilde{s}}} (\log P(s, x|\theta) - \log P(x|\theta)) \quad (2)$$

where the second part was already solved in (1), and the first part, after algebraic manipulations, can be proved to be equal to:

$$\frac{\partial}{\partial \theta_{\tilde{x}, \tilde{s}}} \log P(s, x|\theta) = \frac{\varsigma(\tilde{x}, \tilde{s})}{\theta_{\tilde{x}, \tilde{s}}} - \varsigma(\tilde{s}) \quad (3)$$

where $\varsigma(\tilde{x}, \tilde{s})$ is the number of times state \tilde{s} is visited and amino acid \tilde{x} is emitted from there in the most likely path, which can be found efficiently using the Viterbi algorithm. Likewise, $\varsigma(\tilde{s})$ is the number of times state \tilde{s} is visited in the optimal (or most likely) path. Plugging (1) and (3) into (2) we obtain the complete definition of the constrained Fisher score:

$$\frac{\partial}{\partial \theta_{\tilde{x}, \tilde{s}}} \log P(s|x, \theta) = \left(\frac{\varsigma(\tilde{x}, \tilde{s})}{\theta_{\tilde{x}, \tilde{s}}} - \varsigma(\tilde{s}) \right) - \left(\frac{\varepsilon(\tilde{x}, \tilde{s})}{\theta_{\tilde{x}, \tilde{s}}} - \varepsilon(\tilde{s}) \right). \quad (4)$$

Inspection of (4) can provide a physical interpretation as to why the constrained Fisher score can potentially outperform the unconstrained score in PPI prediction tasks. Notice the ε 's in (4), being expected values, are concerned about the statistical behavior of the dependency of the model on parameter $\theta_{\tilde{x}, \tilde{s}}$, whereas the ς 's, which are actual counts, deal with the explicitly observed data points (or sequences) and the relevance of $\theta_{\tilde{x}, \tilde{s}}$ on their alignment to the model, not caring about the probabilistic behavior. In this sense, the constrained model reduces the biasing of the unconstrained model, and in consequence it directly enhances the information entropy. Improving the entropy of the model means reducing the amount of prior information built into the distribution. The final result is a better learning algorithm.

3 Data

3.1 The Database of 3D Interacting Domains (3DID)

It is known that existing information on the atomic structure of protein complexes can shed light on molecular details necessary for understanding how interactions occur. Since proteins are composed of modular elements (domains) that

to a great extent determine their structure, function and interaction partners, it is desirable that relevant information is organized around domains rather than full-length proteins. To this end, a relational database of 3D Interacting Domains (3DID) was created by Stein *et.al* [9] that contains a collection of domain-domain interactions in proteins for which high-resolution three-dimensional structures are known.

The version of 3DID downloaded for this study has 3,034 pairs of domain-domain interactions, where the 3D structure of every interaction pair is known to decide if the pair physically interact. The 3DID criterion for physical interactions requires that at least five contacts (hydrogen bonds, electrostatic or van der Waals interactions) between the two domains have been detected. Knowing the protein complexes at the atomic level, 3DID is able to provide information on which amino acids actually take part on the interaction. This knowledge allows us to construct an interaction profile hidden Markov domain (ipHMM) to characterize each domain in 3DID by training the model on all sequences that contain the domain using the procedure described in [7]. Consequently, each domain-domain interaction is now characterized by the two respective profile hidden Markov model for the two domains.

3.2 Preparation of Training and Testing Sets

In order to test our method, we wanted to use DDIs that could produce datasets with a sufficient number of (positive and negative) examples for reliable training and testing. Specifically, we selected 94 domain-domain interactions (DDI) with more than 5 positive and negative examples, and where the domain length (number of match states) is smaller than 105. For each DDI, there are two domains, *Dom. A* and *Dom. B*, and *I* pairs of proteins that have been found to physically interact. Each protein in the pair contains either *Dom. A* or *Dom. B*. For every protein sequence that is part of a single DDI, the 3DID database provides a binary vector with the same length of the protein and where the 1's indicate interacting amino acids. These vectors and the profile hidden Markov models of each domain, extracted from Pfam [7], are used to create interacting profile hidden Markov (ipHMM) models for both domains.

To illustrate how we prepare the training and testing datasets, let us take a single DDI as an example. First, all the protein sequences in the DDI are aligned to their corresponding ipHMM. Two different types of alignments are calculated for each sequence: one alignment is obtained by using the posterior decoding (forward and backward algorithm) and the other alignment by using the Viterbi algorithm. As explained in Sect. 2.2, only the former is needed to calculate the unconstrained Fisher vectors, but for the constrained vectors the two alignments are needed. Both alignments can be efficiently calculated through dynamic programming. As a result, each protein sequence can be numerically represented by two different vectors: the constrained Fisher score vector and the unconstrained Fisher score vector. We will later show the constrained vector is advantageous for predicting interacting pairs.

Table 1. Characteristics of the tested dataset, averaged per domain-domain interaction

	Avg. over 94 DDIs
Domain length:	71.9
# of protein pairs:	16.3
# of positive examples:	13.2
# of negative examples:	80.1

Positives examples are constructed by concatenating the Fisher vectors of interacting protein pairs, whereas negative examples are pairs of proteins that have not been observed to interact despite of containing the interacting domains. We found that the 3DID dataset is highly redundant, in the sense that protein sequences at each side of the DDI can be repeated. This redundancy occurs because the sequences that we work with are not the complete protein chains, but only the segments that include the domain. For this reason, every potential positive is included in the dataset only if it is not an already constructed positive. In a similar way, a negative is included only if it is not a positive and it has not been already included in the negative set.

By applying this procedure, we are able to create a set of positive and negative examples for each of the 94 DDIs. Table 1 shows relevant characteristics of the entire dataset, averaged over the 94 DDIs that were used in this study. Ideally, with a non-redundant dataset, the number of positive examples per DDI would be the number of protein pairs in the interaction (I), and the number of negative examples would be $I^2 - I$. But since we have to carry out the described “cleaning”, the number of positives and negatives is reduced, as shown in the table. A “leave one out” (LOO) strategy is followed for testing and training, to guarantee that each positive gets to be predicted and evaluated. Specifically, a test dataset is taken apart by including a single positive example and as many negative examples as needed to guarantee that the relation of positives to negatives in the original dataset is maintained. The remaining positive and negative examples will form a training set that is fed into a support vector machine (SVM) classifier.

4 Results

In this study, the support vector machine package SVMLight [8] is used to predict interacting pairs on the test set. Gaussian kernel with default parameters are used. The number of iterative training and testing stages that are run per DDI is equal to the number of positive examples in the DDI. The prediction results are averaged over all the iterations. Please note that each DDI is independently trained and tested.

A protein’s Fisher vector has 20 (number of amino acids) times the domain’s length (number of match states) positions. Given the numbers in Table 1, we can estimate that a Fisher vector has around 1,440 positions, and since positive and negative examples come from the concatenation of two Fisher vectors,

the approximated average length of an example is 2,870. Considering the LOO strategy being followed for training and testing, this vector length can cause prohibitive running times if the goal is to run a considerable number of DDIs. To address this issue, we adopted dimensionality reduction by means of the singular value decomposition (SVD).

Define the positive dataset as PO , a matrix of size $p \times l$, where p is the number of positive examples, and l is the length of each example vector. Likewise, NE , of size $n \times l$, defines the negative dataset. Let us focus on PO for the time being. The SVD of PO attempts to find two sets of orthonormal vectors, $\{\hat{v}_1, \hat{v}_2, \dots, \hat{v}_r\}$ and $\{\hat{u}_1, \hat{u}_2, \dots, \hat{u}_r\}$, where the former is a basis for the row space PO , and the later is a basis for the column space of PO . Also, r is the rank of PO , \hat{v}_i is of length p and \hat{u}_i is of length l . If we define $V_{PO} = [\hat{v}_1 \hat{v}_2 \dots \hat{v}_r]$ and $U_{PO} = [\hat{u}_1 \hat{u}_2 \dots \hat{u}_r]$, it is possible to show that $PO \cdot V_{PO} = U_{PO} \cdot S$, where S is a diagonal matrix that contains the so called “singular values” of the decomposition. Since V_{PO} is orthonormal, the positive dataset can be factorized as $PO = U_{PO} \cdot S \cdot V_{PO}^T$, where U_{PO} is of size $p \times r$ and V_{PO} is of size $l \times r$. The bases for the row space of PO provide a means to reduce the dimensionality of the positive examples in the following way: If we create a new matrix V_{PO}^{RED} (RED stands for “reduced”) of size $l \times k$, where $k < r$ (only the strongest k base vectors have been maintained), we could project our positive dataset onto V_{PO}^{RED} by simply doing $PO^{RED} = PO \cdot V_{PO}^{RED}$. Clearly, PO^{RED} is of size $p \times k$. In other words, PO^{RED} is a new positive dataset that contains the same number of examples as PO , but where the dimensionality of each example has been reduced to k . An identical procedure is applied to the negative dataset as well.

In order to find a desirable value for k , we randomly chose a domain-domain interaction from 3DID, and created the Fisher vectors for that DDI in all the scenarios being considered in this study (non-interacting and interacting match states, unconstrained and constrained vectors). We calculated the singular value decomposition of the positive and the negative datasets in each case. The singular values of the decomposition can be used to get an idea of the number of singular vectors that concentrate the information contained in the dataset. The randomly chosen DDI consists of a 71 amino acids long domain that binds to itself ($Dom. A = Dom. B$). This is the b1 domain (Pfam PF02246) found in protein L, which is a bacterial protein with immunoglobulin (Ig) light chain-binding properties. Figures “Singular Values Non Int Match States.eps” and “Singular Values Int Match States.eps”, available online at http://liao.cis.udel.edu/CFisher_for_PPI_prediction/, show that in the four scenarios, $k = 10$ suffices to catch most of the energy in the datasets. In these figures, positive and negative singular values have been averaged.

Having decided on $k = 10$, we tested all the DDIs in our dataset using after applying dimensionality reduction. The averaged prediction results are summarized in Table 2. The ROC curves corresponding to this table were plotted in Fig. 2, where we also show the performance of a baseline model, as reported in Patel and Liao, in which the vectors are constructed using the log-likelihood score of the alignment of each sequence to the two domains involved in the interaction.

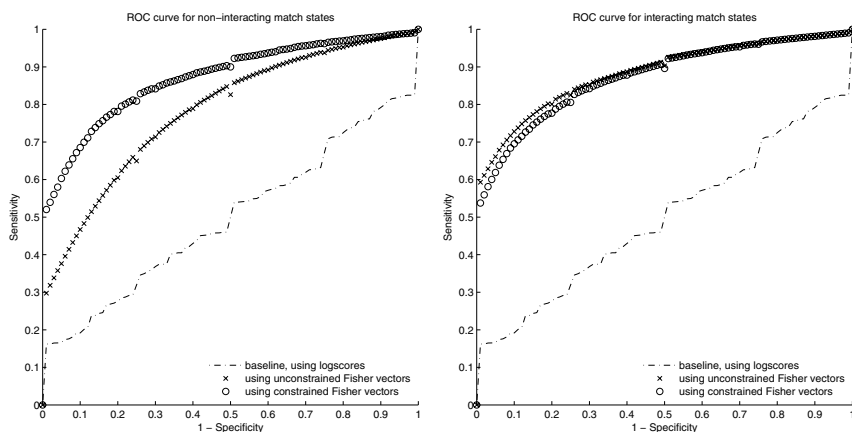


Fig. 2. ROC curves. Comparison between unconstrained and constrained Fisher vectors on non-interacting (left) and interacting (right) match states.

The left side of Fig. 2 shows how the constrained Fisher vectors improve the performance of the classifier over the unconstrained approach, when calculating the Fisher scores against non-interacting match states. The first row of Table 2 reveals an improvement of 17%, from $\text{ROC} = 0.7012$ to $\text{ROC} = 0.8206$. On the other hand, when the Fisher scores are calculated against interacting match states, both the right side of Fig. 2 and the second row of Table 2 show the performance is hardly affected when we switch from the unconstrained to the constrained Fisher scores. Interestingly enough, these results agree with our reasoning at the end of Sect. 2.3, where we anticipated that the constrained Fisher vector has potential to enhance the entropy of the model being learned by reducing the information that is injected into the trained SVM by the observed samples of the distribution. As compared to the interacting match states, the non-interacting match states are more of a "background", and the constrained Fisher vector would be more informative as it subtracts this background out.

Finally, we tested the influence that the dimensionality reduction of Fisher vectors based on the singular value decomposition has on the performance of the support vector machine predictor. We had shown that, for a randomly chosen DDI, the first 10 components of the decomposition are able to pick up most of the dataset's information. Now, for the same DDI, we run protein-protein interaction predictions following the LOO strategy described before. The results

Table 2. Comparison of ROC scores using unconstrained and constrained Fisher vectors of length 10

	unconstrained	constrained
Non-interacting	0.701	0.821
Interacting	0.805	0.800

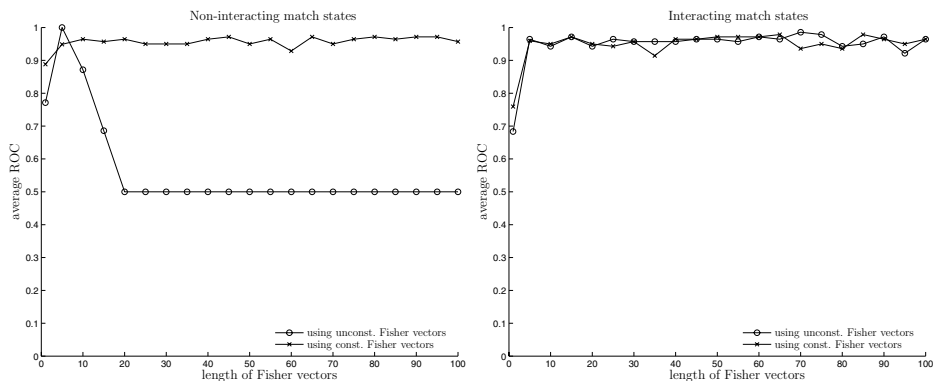


Fig. 3. Fisher vectors were calculated with respect to non-interacting (left) and interacting (right) match states. The curves show prediction results after varying k , the length of the Fisher vectors.

are shown in Fig. 3. Although it would have been desirable to run this experiment on all the DDIs in the dataset, this single interaction already reveals interesting results. When using Fisher scores against non-interacting match states (left side of Fig. 3), the constrained approach outperforms the unconstrained for almost all values of k . Note in this figure how the predictor based on unconstrained Fisher vectors loses its learning ability when $k > 20$. Besides, in all other scenarios, the learning ability of the predictor is not improved by increasing k . As we had anticipated, $k = 10$ is able to capture the most relevant information of the model. Remarkably, a dimensionality reduction that was originally adopted to speed up the model training ended up improving the prediction results as well.

Note that while the dimension reduction (or rather feature selection) from SVD is proved to be very effective, it relies on using separate projection vectors V 's for the positive and negative examples. This limits its predictive utility when an example is not known a priori as positive or negative, though the limitation can be largely alleviated by using some standard feature selection techniques.

5 Conclusion

In this work, we developed a method based on a framework that is capable of combining generative models and discriminative classifiers. By leveraging the interaction profile hidden Markov models trained on interacting protein domains whose structure is known, we are able to transfer the domain structural information to proteins that lack such information. The constrained Fisher scores can further extract more domain specific information by conditioning on the most probable (Viterbi) path aligning protein sequences to the ipHMM, and thus form into more informative feature vectors representing protein pairs that can be more easily classified by the support vector machine. As demonstrated in cross validation experiments, the prediction accuracy increases significantly. As future work, we will focus on integrating feature selection mechanisms with

the learning process within a semi-supervised learning framework so that the method can be more efficiently applied to genome wide prediction even with limited training data.

Acknowledgements

The authors are grateful to Tobias Muller and Birgit Pils for making available their data and Matlab implementation of ipHMM. The authors would also like to thank Tapan Patel for helping with processing the 3DID database. Finally, the authors like to thank the anonymous reviewers for their helpful comments.

References

1. Rabiner, L.R., Juang, B.H.: An introduction to hidden Markov models. *IEEE ASSP Magazine* 3(1), 4–16 (1986)
2. Jaakkola, T.S., Haussler, D.: Exploiting generative models in discriminative classifiers. In: *Advances in Neural Information Processing Systems*, vol. 11, pp. 487–493. MIT Press, Cambridge (1998)
3. Jaakkola, T.S., Diekhans, M., Haussler, D.: A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*, 95–114 (2000)
4. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: *Biological sequence analysis*. Cambridge University Press, Cambridge (1998)
5. Patel, T., Liao, L.: Predicting protein-protein interaction using Fisher scores extracted from domain profiles. In: *Proceedings of IEEE 7th International Symposium for Bioinformatics and Bioengineering (BIBE)*, Boston, MA (2007)
6. Kahsay, R., Gao, G., Liao, L.: Discriminating Transmembrane Proteins From Signal Peptides Using SVM-Fisher Approach. In: *The Proceedings of The Fourth International Conference on Machine Learning and Applications (ICMLA 2005)*, Los Angeles, CA, pp. 151–155 (2005)
7. Finn, R., Mistry, J., Schuster-Böckler, B., Griffiths-Jones, S., Hollich, V., Lassmann, T., Moxon, S., Marshall, M., Khanna, A., Durbin, R., Eddy, S., Sonnhammer, E., Bateman, A.: Pfam: clans, web tools and services. *Nucleic Acids Research (Database Issue)* 34, D247–D251 (2006)
8. Joachims, T., Scholkopf, B., Burges, C., Smola, A.: *Making large-Scale SVM Learning Practical*. *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge (1999)
9. Stein, A., Russell, R., Aloy, P.: 3did: interacting protein domains of known three-dimensional structure. *Nucleic Acids Research* 33(Database issue), D413–D417 (2005)
10. Itzhaki, Z., Akiva, E., Altuvia, Y., Margalit, H.: Evolutionary conservation of domain-domain interactions. *Genome Biology* 7, R125 (2006)
11. Friedrich, T., Pils, B., Dandekar, T., Schultz, J., Muller, T.: Modeling Interaction Sites in Protein Domains with Interaction Profile Hidden Markov Models. *Bioinformatics* 22, 2851–2857 (2006)

Improving Protein Localization Prediction Using Amino Acid Group Based Physicochemical Encoding

Jianjun Hu* and Fan Zhang

Department of Computer Science and Engineering, University of South Carolina, Columbia,
SC, 29208, USA

{jianjunh, zhangf}@cec.sc.edu

Abstract. Computational prediction of protein localization is one common way to characterize the functions of newly sequenced proteins. Sequence features such as amino acid (AA) composition have been widely used for subcellular localization prediction due to their simplicity while suffering from low coverage and low prediction accuracy. We present a physicochemical encoding method that maps protein sequences into feature vectors composed of the locations and lengths of amino acid groups (AAGs) with similar physicochemical properties. This high-level modular representation of protein sequences overcomes the shortcoming of losing order information in the commonly used AA composition and AA pair composition encoding. When applied with SVM classifiers, we showed that AAG based features are able to achieve higher prediction accuracy (up to 20% improvement) than the widely used AA composition and AA pair composition to differentiate proteins of different localizations. When AAGs and AA composition encoding combined, the prediction accuracy can be further improved thus achieving synergistic effect.

Keywords: Physical encoding, protein subcellular location prediction, AA index, Support Vector Machines, amino acid groups.

1 Introduction

Determination of subcellular locations of a protein experimentally [1;2] or computationally [3-6] can greatly help to infer its function. Due to its simplicity, automated prediction of subcellular localization has been routinely used to annotate protein sequences and dozens of algorithms have been developed [3-6]. These algorithms employ a variety of supervised machine learning techniques including neural networks [7;8], nearest neighbor classifier, Markov models, Bayesian networks [9], expert rules, meta-classifiers [10], and the support vector machines [11-13]. While algorithm variation can tune up the prediction performance, a more critical factor for accurate prediction is to extract effective features for inferring the subcellular location of a protein. A variety of information has been used as features for subcellular prediction as discussed below.

* Correspondence Author.

1.1 Sequence Based Features for Protein Localization Prediction

Predicting protein localization from sequences only is a holy grail of protein localization annotation. This requires detailed understanding of how protein sorting signals are encoded and recognized. There have been many sequence based prediction algorithms, which can be classified into the following categories.

1) Signal motif based models such as those used in pSort [14] and TargetP [8]: these features are based on the signal hypothesis of protein sorting and use signal motifs in the N-terminal portion of a protein. Currently, only secretory signals are well characterized by signal motif models and can be predicted with high precision. These features are not as successful for other locations.

2) Amino acid (AA) and amino acid pair (PairAA) composition [15-19]: These features are based on the bias of protein compositions for different locations and have been widely used in previous work due to its simplicity and high coverage. Usually, the whole sequence of a protein is used to define the AA or PairAA features. A limitation of AA or PairAA features is that they do not consider order information within protein sequences and the prediction accuracy based on them is limited.

3) Pseudo AA composition and gapped amino acid composition [15]: these features try to capture the order information by considering the correlation relationships of neighbor amino acids with different gaps in terms of physicochemical properties such as hydrophobicity, hydrophilicity, and mass.

1.2 Functional Annotation Based Features for Localization Prediction

Subcellular location prediction algorithms have also been developed that take advantage of localization information of annotated proteins with indirect relationships with the query protein. This includes functional annotation [20], phylogenetic profiling [21], homology [22], and protein-protein interaction [23] and etc. Algorithms that integrate multiple sources of information such as Drawid et al.'s naïve Bayesian predictor [24] uses signal motifs, gene expression patterns, and overall-sequence properties. Scott et. al.'s Bayesian network predictor [9] incorporates protein motifs, targeting signals, and protein-protein interaction data. While these methods can achieve better performances in terms of localization prediction, they contribute little to the understanding of the protein sorting mechanisms.

In this paper, we propose a physicochemical encoding method (AAG) that converts protein sequences into high-level feature vectors representing the modular amino acid group structures such as hydrophobicity or positive charge groups. Compared to amino acid composition, our encoding captures the order information of the amino acid groups and aligns well with sorting signals. Compared to the pseudo amino acid composition, the AAG encoding can capture the global orders of the amino acid groups within protein sequences.

2 Materials and Methods

We developed an amino acid index based physicochemical encoding (AAG Coding) for proteins and applied it to protein localization prediction using a support vector machine classifier. We applied the algorithm to a reduced non-redundant protein sequence set used by Bacello algorithm for comparing protein localization predictors [3].

Most protein sorting signals are encoded in the amino acid sequences of proteins and have block-like structures with different physicochemical properties such as hydrophobicity, charge, and alpha structures. We developed a OneScan algorithm to detect these amino acid groups (AAGs), subsequences with homogeneous physicochemical properties using their amino acid index [25] scores, from the whole sequences. Each sequence will be transformed into a 1088-dimension feature vector composed of the first positions of AAGs and their lengths. We then apply support vector machines (SVM) algorithm to differentiate protein sequences among pairs of locations.

2.1 Modular Structure of Protein Sorting Signals

Protein targeting signals are amino acid sequences responsible for directing proteins to their target locations. They are usually located at the N-terminal or C-terminal of a protein sequence [26]. Compared to DNA motifs which are conserved at the nucleotide sequence level, protein targeting signals are much less conserved at the amino acid level [27]. They usually have high variation in length and type. Despite this high variation among signaling motifs targeting the same location, the cell can recognize all kinds of targeting signals with almost 100% selectivity and specificity [28]. This discrepancy implies that conserved patterns of targeting signals are not reflected at the amino acid level, but at the level of physicochemical properties such as the hydrophobic regions, polar regions, alpha-helix regions, and hydroxylated regions [29]. The modular structure of sorting signals have been experimentally confirmed for signals of extracellular, mitochondrial, chloroplast, thylakoid lumen, peroxisome proteins and etc (Figure 1). For example, secretory targeting signals [8], usually contain a positively charged N-terminal domain, a hydrophobic h-region, and a cleavage site region with mostly small residues. Each region consists of a couple of consecutive amino acids sharing common properties. We denote these regions as amino acid groups (AAG), the building blocks of targeting signals.

Given an amino acid sequence, we can define many high-level amino acid properties [30], some of which are involved in targeting signal recognition. Examples include bias of amino acids, hydrophobicity, polarity, and etc. These properties are summarized by the comprehensive amino acid index database (AAIndex) [25], which has already been employed for protein classification, protein localization prediction, and specific protein type recognition. We use 544 amino acid indexes of the AAindex [25] database to represent high-level physicochemical and other high-level properties. Each index is a set of 20 numerical values representing various physicochemical and biochemical properties of the 20 amino acids. These properties can be roughly classified into four major clusters: 1) α -helix and turn propensities, 2) β -strand propensity, 3) hydrophobicity, and 4) physicochemical properties such as charge, solvent accessibility, and polarity. Recognition of protein targeting motifs strongly depends on these properties of amino acid segments within targeting signal sequences. Actually, it is suggested that the size, charge, and hydrophobicity are the three fundamental biophysical properties of amino acids that determines their bioactivity [31;32].

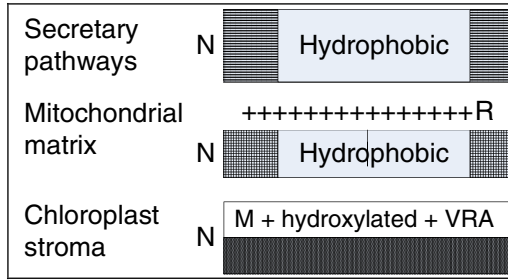


Fig. 1. Modular structures of protein targeting signals. Patterns correspond to amino acid groups with similar physichemical properties.

2.2 Physichemical Encoding Using Amino Acid Index Groups

To represent modular protein targeting signals at the physichemical levels, we will design an encoding scheme to convert protein amino acid sequences into physical property vectors. Instead of calculating correlations of a few physical properties of all the possible pairwise amino acids with k gaps as the pseudo amino acid composition method does [15], AAG coding aims to capture modular structure of sorting signals and also the global order of the component amino acid groups such as hydrophobicity or positive charge AAGs. The encoding process is composed of the following steps:

- (1) For each AA index R , convert protein sequences into amino acid index vector

In the AA index database, hydrophobicity index JOND750101 is defined by the following scores, each amino acid has a corresponding hydrophobicity value.

A/L	R/K	N/M	D/F	C/P	Q/S	E/T	G/W	H/Y	I/V
0.87	0.85	0.09	0.66	1.52	0.00	0.67	0.10	0.87	3.15
2.17	1.64	1.67	2.87	2.77	0.07	0.07	3.77	2.67	1.87

Given a protein sequence $p = \{A_1, A_2, \dots, A_n\}$, we will replace the amino acids with its corresponding index values and generate the index score vector $s = \{s_1, s_2, \dots, s_n\}$.

- (2) Label amino acids with significant higher index score than average.

To identify significant AAG groups w.r.t. AA index R, the average μ and standard deviation σ of the vector $s = \{s_1, s_2, \dots, s_n\}$ is calculated. Then for a given amino acid A_i with index score s_i , if $s_i \geq \mu + \lambda\sigma$ where λ is a parameter, we will label A_i as a position with unusual physichemical property such as hydrophobicity, denoted as $L_i = 1$. Similarly, if $s_i \leq \mu - \lambda\sigma$, then $L_i = -1$, otherwise $L_i = 0$. The reason to divide amino acid attribute values into three groups

for an AA index is that indexes such as charge and secondary structure can only have three categories and thus three groups is a good choice overall for all AA indexes. This method has been used by PROFEAT [30], a web server for computing structural and physicochemical features of proteins from amino acid sequences. With this mapping, the protein sequence DCENQTYSHGKR will be mapped to an AA index sequence $R_{-1}R_0R_{-1}R_0R_0R_0R_0R_0R_{+1}R_0R_{+1}R_{+1}$, where R_{+} , R_0 , R_{-} corresponds to positive, neutral, and negative charges of corresponding amino acids.

- (3) Merge neighboring amino acids of index sequences with positive or negative labels into AAG groups if they have the same label. So for previous sequence example, we will have $\overline{R_{-1}R_0} \overline{R_{-1}R_0R_0R_0R_0} \overline{R_{+1}R_0} \overline{R_{+1}R_{+1}}$. To give some tolerance to small gaps, we will keep merging two positive or negative AAGs if the gap between them is smaller than a given threshold value, e.g. $G_0 = 1$. So we will have $\overline{R_{-1}R_0R_{-1}R_0R_0R_0R_0} \overline{R_{+1}R_0R_{+1}R_{+1}}$. There are two R index AAG groups. Considering that in protein sorting process, a minimum number of amino acids are needed to compose a meaningful physicochemical AAG groups, we only keep AAGs that have the minimum length, e.g. $L_{\min} = 4$. In this case, we obtain one AAG for AA index R , denoted as $AAG_R = \{pos, len\} = \{8, 4\}$, where pos is the starting position of the AAG and len is its length.
- (4) Assemble the AAG feature vector. For each AA index, we may find multiple positive or negative AAGs. In this paper, we only keep the first positive AAG groups for each AA index. Since we have 544 AA indexes, each with one positive AAG represented by two values, together we will have a 1088 component feature vector which is used to represent the physicochemical properties of the proteins.

2.3 Protein Localization Classification Using One-vs-One SVM with Feature Selection

Both Loctree [33] and Bacello [34], two of the leading protein localization algorithms use a decision tree of binary SVM classifier to make prediction. In this paper, binary SVM classifiers with linear kernels are used to differentiate the following pairs of proteins: Secretary, Mitochondrial, Nucleic, and cytoplasmic. We use the libsvm SVM classifier wrapped in the Weka data mining system [35].

Compared to the Loctree and Bacello where full raw features are used to train the SVM classifiers, we applied a feature selection preprocessing to remove redundant attributes calculated from AAG detection algorithms or AA composition counting. The idea is that the 544 AA indexes in the AA index database are redundant and feature selection can pick those relevant features that contribute to the classification of the proteins of two different locations.

3 Results and Discussion

We use two sets of datasets and a series of experiments to demonstrate the advantages of AAG encoding for differentiating proteins of different locations. The binary SVM classifier is used to ensure that the classification results are not complicated by the complex multi-class SVMs that are usually used for protein localization prediction.

3.1 Data Preparation

We used the following two eukaryotic benchmark datasets used in comparison study of protein localization algorithms [36;37]. The first dataset is the reduced dataset which is obtained by first collecting newly added eukaryotic proteins after the release 48 of SwissProt database. Then a 30% homogeneity reduction is applied to ensure any pair of sequences has maximum identify of 30%. The second dataset is all sequences extracted from SwissProt up to version SwissProt 41, again with a maximum 30% homogeneity reduction. The numbers of sequences in each dataset are below:

Seq. No. of Datasets	Cytoplasm	Nucleic	Mitochondrial	Secretary
Reduced set (DR)	104	273	60	139
Train set (DT)	302	803	153	632

We also compared the performances on a Balanced DR dataset in which for each pair of locations, we use sub-sampling to reduce the number of sequences of the majority class to the number of sequences of the class with less sequences.

We use a 10-fold cross validation to measure the performance of the training binary SVM classifiers with different features. This ensures that our results are not biased due to our selection of the training data.

3.2 Comparing the Performance of AAG Physical Encoding with AA Composition

We compared the SVM classification performance of AAG encoding with amino acid encoding with 20 amino acid components and 400 pair-AA components, totaling 420 features created from the whole sequence. 10-fold cross-validation is used to obtain unbiased evaluation on the DT dataset. Similar results were achieved when 5-fold cross-validation is used.

Table 1 shows that the prediction accuracy for separating proteins of two locations among nucleic, mitochondrial, cytoplasmic and secretary proteins. The AAG encoding here is applied with minimum AAG width of 5. Out of the 6 pairwise classifications, the AAG encoding has significant improvement in terms of accuracy (from 8% to 20%) for separating secretary proteins from other proteins. It also has moderate improvement for separating mitochondrial proteins from cytoplasmic and nucleic proteins. The only performance decrease is for classifying Nucleic-Cytoplasmic proteins. The maximum improvement for separating secretary proteins from cytoplasmic proteins is not unexpected as most secretary proteins have the modular structures that motivate AAG encoding. The worse performance for nucleic and cytoplasmic proteins implies that nucleic protein sorting signals may be more intricate or does not

Table 1. Comparison of classification accuracy of AA composition and AAG encoding applied to the DT dataset. 10-fold cross-validation is applied. M-Mitochondrial; C-Cytoplasmic; N-Nucleic; S-Secretary.

	M-C	M-N	N-C	S-C	S-M	S-N
AA420	75.28	86.34	70.28	70.96	80.34	79.14
AAG	78.30	88.77	61.55	85.76	88.00	86.09
AAG+AA420	83.09	92.1	68.66	88.17	92.14	87.67

contain well-defined modular AAG groups. When AAG and AA composition encoding are combined, we can achieve even larger performance gain for up to 23% for secretary-cytoplasmic protein classification.

3.3 Effect of Balanced or Imbalanced Datasets on AA composition and Physical Encoding

It was reported that imbalanced datasets may have big influence on SVM based classification performance. We applied the AA composition encoding with 420 features and AAG encoding with 1088 features to the balanced and imbalanced DR dataset. Table 2 shows that both encoding methods are affected by the testing datasets with lower performance for balanced dataset partially due to the fact that balanced datasets have much fewer sequences after sub-sampling. But in general, AA encoding has much smaller decrease of the prediction accuracy for distinguishing secretary proteins from other proteins. Again, it is shown that the combined AAG and AA420 encoding has the best prediction accuracy.

Table 2. Comparison of classification accuracy of AA composition and AAG encoding applied to the balanced and imbalanced DR datasets. 10-fold cross-validation is applied. M-Mitochondrial; C-Cytoplasmic; N-Nucleic; S-Secretary.

	M-C	M-N	N-C	S-C	S-M	S-N
AA420	87.20	91.29	65.52	82.30	85.43	89.32
AA420(B)	79.66	83.05	61.17	80.58	80.51	85.14
AAG	85.36	93.09	67.10	93.41	90.45	96.84
AAG(B)	75.42	89.93	56.31	93.20	88.13	96.66
AAG+AA420	89.02	93.39	72.41	94.23	91.95	97.08
AAG+AA420(B)	82.2	94.92	61.65	95.63	88.98	96.01

3.4 Effect of Feature Selection on Classification Performance

With 420 and 1180 features for AA420 and AAG encodings respectively, it is interesting to see whether feature selection can be used to achieve better performance. Here we applied a supervised feature selection with best-first search scheme in Weka system [35] to choose features. In this method, Subsets of features that are highly correlated with the class while having low intercorrelation are preferred [38]. In the cross-validation experiments below, for each training-testing dataset pair, we run the feature selection on the training datasets first to determine the feature subset and then use this feature subset for training and testing.

Table 3 compares the performances of the AA composition, AAG encoding, and AA+AAG with DT dataset with or without feature selection. It shows that for AA420 encoding, feature selection can improve the performance a little bit up to 6% for 5 out of 6 binary classifications. Similarly, AAG can benefit from feature selection for some protein classifications with almost 19% improvement for distinguishing Nucleic-Cytoplasmic. Using the combined AA composition and AAG encoding together with feature selection, we achieve only better prediction accuracy for separating the most difficulty types of proteins nucleic and cytoplasmic, improving from 68.66% to 81.65%. On the other hand, for other location pairs, feature selection generates worse results. In summary, the improvement of feature selection on most binary SVM classifications here are limited or worse since SVM has inherent better feature selection capability than the standalone supervised feature selection.

Table 3. Comparison of classification accuracy of AA composition and AAG encoding applied to dataset DT dataset with or without feature selection. 10-fold cross-validation is applied. M-Mitochondrial; C-Cytoplasmic; N-Nucleic; S-Secretary.

	M-C	M-N	N-C	S-C	S-M	S-N
AA420	75.28	86.34	70.28	70.96	80.34	79.14
AA420(FS)	76.07	87.44	74.57	72.72	80.64	78.62
AAG	78.30	88.77	61.55	85.76	88.00	86.09
AAG(FS)	82.93	90.02	73.33	84.63	86.89	88.07
AAG+AA420	83.09	92.1	68.66	88.17	92.14	87.67
AAG+AA420(FS)	81.65	81.65	81.65	81.65	81.65	81.65

4 Discussion and Conclusion

We proposed a new protein sequence encoding method inspired by the modular structures of protein sorting signals. The amino acid index based AAGs feature allows us to capture the internal signals as well as the composition bias of proteins in different subcellular locations. It was shown that compared to the widely used amino acid composition AA and AAPair, AAG encoded features can achieve up to 20% improvement in terms of prediction accuracy for differentiating proteins of different subcellular locations and is especially effective for differentiating secretory proteins from other proteins.

Our AAG encoding differs from the digital coding proposed [39] since we allow overlapped AAGs and our AAGs are composed of multiple amino acids. AAG encoding is also different from the pseudo amino acid composition (PseAA) [15] which also aims to capture the order information of the amino acids and the physicochemical correlation of neighboring amino acids. However, the major difference is that PseAA does not capture the global order of different amino acid groups with different physicochemical properties while AAG encodes this information.

The success of AAGs implies that the AAGs extracted in our algorithm can be further systematically mined using pattern mining algorithms so that novel signal models similar to canonical secretory signals can be discovered for many other subcellular locations or for different secretory pathways. This paper has only reported the

prediction results for binary classification. Extension of the binary SVM classifiers to multi-class SVM classifiers is underway and will be reported elsewhere in which performance comparison will be evaluated and compared with other algorithms. The encoding program of AAG can be freely downloaded from <http://mleg.cse.sc.edu/aag>

Acknowledgments. This work is partially supported by the National Science Foundation/EPSCoR under Grant No. EPS-0447660.

References

1. Huh, W.K., Falvo, J.V., Gerke, L.C., Carroll, A.S., Howson, R.W., Weissman, J.S., O'Shea, E.K.: Global analysis of protein localization in budding yeast. *Nature* 425, 686–691 (2003)
2. Kumar, A., Agarwal, S., Heyman, J.A., Matson, S., Heidtman, M., Piccirillo, S., Umansky, L., Drawid, A., Jansen, R., Liu, Y., et al.: Subcellular localization of the yeast proteome. *Genes Dev.* 16, 707–719 (2002)
3. Casadio, R., Martelli, P.L., Pierleoni, A.: The prediction of protein subcellular localization from sequence: a shortcut to functional genome annotation. *Brief Funct. Genomic. Proteomic.* 7, 63–73 (2008)
4. Emanuelsson, O., Brunak, S., von Heijne, G., Nielsen, H.: Locating proteins in the cell using TargetP, SignalP and related tools. *Nature Protocols* 2, 953–971 (2007)
5. Gardy, J.L., Brinkman, F.S.L.: Methods for predicting bacterial protein subcellular localization. *Nature Reviews Microbiology* 4, 741–751 (2006)
6. Sprenger, J., Fink, J.L., Teasdale, R.D.: Evaluation and comparison of mammalian subcellular localization prediction methods. *Bmc Bioinformatics* 7 (2006)
7. Shen, H.B., Yang, J., Chou, K.C.: Methodology development for predicting subcellular localization and other attributes of proteins. *Expert Review of Proteomics* 4, 453–463 (2007)
8. Emanuelsson, O., Nielsen, H., Brunak, S., von Heijne, G.: Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *Journal of Molecular Biology* 300, 1005–1016 (2000)
9. Scott, M.S., Calafell, S.J., Thomas, D.Y., Hallett, M.T.: Refining protein subcellular localization. *PLoS Comput. Biol.* 1, 518–528 (2005)
10. Jin, Y.H., Niu, B., Feng, K.Y., Lu, W.C., Cai, Y.D., Li, G.Z.: Predicting subcellular localization with AdaBoost Learner. *Protein and Peptide Letters* 15, 286–289 (2008)
11. Lorena, A.C., de Carvalho, A.C.P.L.: Protein cellular localization prediction with support vector machines and decision trees. *Computers in Biology and Medicine* 37, 115–125 (2007)
12. Sarda, D., Chua, G.H., Li, K.B., Krishnan, A.: pSLIP: SVM based protein subcellular localization prediction using multiple physicochemical properties. *Bmc Bioinformatics* 6 (2005)
13. Hua, S.J., Sun, Z.R.: Support vector machine approach for protein subcellular localization prediction. *Bioinformatics* 17, 721–728 (2001)
14. Nakai, K., Horton, P.: PSORT: a program for detecting sorting signals in proteins and predicting their subcellular localization. *Trends in Biochemical Sciences* 24, 34–35 (1999)
15. Chou, K.C., Cai, Y.D.: Predicting subcellular localization of proteins by hybridizing functional domain composition and pseudo-amino acid composition. *Journal of Cellular Biochemistry* 91, 1197–1203 (2004)

16. Nanni, L., Lumini, A.: Genetic programming for creating Chou's pseudo amino acid based features for submitochondria localization. *Amino Acids* 34, 653–660 (2008)
17. Li, Y.F., Liu, J.: Predicting subcellular localization of proteins using support vector machine with N-terminal amino composition. *Proceedings of Advanced Data Mining and Applications* 3584, 618–625 (2005)
18. Shi, J.Y., Zhang, S.W., Pan, Q., Cheng, Y.M., Xie, J.: Prediction of protein subcellular localization by support vector machines using multi-scale energy and pseudo amino acid composition. *Amino Acids* 33, 69–74 (2007)
19. Yu, C.S., Lin, C.J., Hwang, J.K.: Predicting subcellular localization of proteins for Gram-negative bacteria by support vector machines based on n-peptide compositions. *Protein Science* 13, 1402–1406 (2004)
20. Szafron, D., Lu, P., Greiner, R., Wishart, D.S., Poulin, B., Eisner, R., Lu, Z., Anvik, J., Macdonell, C., Fyshe, A., et al.: Proteome Analyst: custom predictions with explanations in a web-based tool for high-throughput proteome annotations. *Nucleic Acids Research* 32, W365–W371 (2004)
21. Marcotte, E.M., Xenarios, I., van der Blik, A.M., Eisenberg, D.: Localizing proteins in the cell from their phylogenetic profiles. *Proceedings of the National Academy of Sciences of the United States of America* 97, 12115–12120 (2000)
22. Yu, C.S., Chen, Y.C., Lu, C.H., Hwang, J.K.: Prediction of protein subcellular localization. *Proteins-Structure Function and Bioinformatics* 64, 643–651 (2006)
23. Zhang, S., Xia, X.F., Shen, J.C., Sun, Z.R.: Eukaryotic protein subcellular localization prediction based on sequence conservation and protein-protein interaction. *Progress in Biochemistry and Biophysics* 35, 531–535 (2008)
24. Drawid, A., Gerstein, M.: A Bayesian system integrating expression data with sequence patterns for localizing proteins: Comprehensive application to the yeast genome. *Journal of Molecular Biology* 301, 1059–1075 (2000)
25. Kawashima, S., Pokarowski, P., Pokarowska, M., Kolinski, A., Katayama, T., Kanehisa, M.: AAindex: amino acid index database, progress report 2008. *Nucleic Acids Research* 36, D202–D205 (2008)
26. Silhavy, T.J., Benson, S.A., Emr, S.D.: Mechanisms of Protein Localization. *Microbiological Reviews* 47, 313–344 (1983)
27. Ng, S.Y.M., Chaban, B., VanDyke, D.J., Jarrell, K.F.: Archaeal signal peptidases. *Microbiology-Sgm* 153, 305–314 (2007)
28. Nielsen, H., Brunak, S., von Heijne, G.: Machine learning approaches for the prediction of signal peptides and other protein sorting signals. *Protein Engineering* 12, 3–9 (1999)
29. Emanuelsson, O.: Predicting protein subcellular localisation from amino acid sequence information. *Brief Bioinform.* 3, 361–376 (2002)
30. Li, Z.R., Lin, H.H., Han, L.Y., Jiang, L., Chen, X., Chen, Y.Z.: PROFEAT: a web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence. *Nucleic Acids Research* 34, W32–W37 (2006)
31. Biro, J.C.: Amino acid size, charge, hydrophathy indices and matrices for protein structure analysis. *Theor. Biol. Med. Model.* 3, 15 (2006)
32. Lu, Y., Bulka, B., Desjardins, M., Freeland, S.J.: Amino acid quantitative structure property relationship database: a web-based platform for quantitative investigations of amino acids. *Protein Engineering Design & Selection* 20, 347–351 (2007)
33. Nair, R., Rost, B.: Mimicking cellular sorting improves prediction of subcellular localization. *Journal of Molecular Biology* 348, 85–100 (2005)
34. Pierleoni, A., Martelli, P.L., Fariselli, P., Casadio, R.: BaCelLo: a balanced subcellular localization predictor. *Bioinformatics* 22, E408–E416 (2006)

35. Witten, I.H., Frank, E.: *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco (2005)
36. Casadio, R., Martelli, P.L., Pierleoni, A.: The prediction of protein subcellular localization from sequence: a shortcut to functional genome annotation. *Brief Funct. Genomic. Proteomic.* 7, 63–73 (2008)
37. Pierleoni, A., Martelli, P.L., Fariselli, P., Casadio, R.: BaCelLo: a balanced subcellular localization predictor. *Bioinformatics* 22, E408–E416 (2006)
38. Hall, M.A., Smith, L.A.: Feature subset selection: a correlation based filter approach. In: *Proceeding of International Conference on Neural Information Processing and Intelligent Information Systems*, pp. 855–858. Springer, Heidelberg (1997)
39. Xiao, X., Chou, K.C.: Digital coding of amino acids based on hydrophobic index. *Protein and Peptide Letters* 14, 871–875 (2007)

The Impact of Gene Selection on Imbalanced Microarray Expression Data

Abu H. M. Kamal, Xingquan Zhu, Abhijit S. Pandya, Sam Hsu,
and Muhammad Shoaib

Department of Computer Science & Engineering, Florida Atlantic University,
Boca Raton, FL 33431, USA
akamall@fau.edu, {xqzhu, pandya, sam}@cse.fau.edu

Abstract. Microarray experiments usually output small volumes but high dimensional data. Selecting a number of genes relevant to the tasks at hand is usually one of the most important steps for the expression data analysis. While numerous researches have demonstrated the effectiveness of gene selection from different perspectives, existing endeavors, unfortunately, ignore the data imbalance reality, where one type of samples (e.g., cancer tissues) may be significantly fewer than the other (e.g., normal tissues). In this paper, we carry out a systematic study to investigate the impact of gene selection on imbalanced microarray data. Our objective is to understand that if gene selection is applied to imbalanced expression data, what kind of consequences it may bring to the final results? For this purpose, we apply five gene selection measures to eleven microarray datasets, and employ four learning methods to build classification models from the data containing selected genes only. Our study will bring important findings and draw numerous conclusions on (1) the impact of gene selection on imbalanced data, and (2) behaviors of different learning methods on the selected data.

1 Introduction

Gene expression data are important sources for many biological tasks such as genetic disease profiling (Golub, et al., 1999), identifying potential biomarkers and signatures for cancers (Xiong et al. 2001), and gene regulatory network reconstruction (Segal et al. 2003). Due to the cost involved in microarray experiments and the availability of the samples, typical microarray experiments have a very small number of samples (e.g., less than 100) but output expression values for a large number of genes (e.g., more than 20,000). Such low volumes but high dimensional data impose significant changes to any data processing algorithms or tools which may possibly apply to the data. For example, the construction of the gene regulatory network relies on the searching of the causal relationships between genes. Such a search process is very time consuming for a large number of genes, and reducing the number of candidate genes is an effective way to improve the efficiency and the accuracy for regulatory network reconstruction. On the other hand, the goal of most microarray experiments is to identify important genes associated to specific diseases and predict the likelihood of a test tissue sample belonging to the disease (Golub, et al., 1999). Most machine

learning classifiers are known to be inaccurate or unstable on high dimensional data, especially when the number of training examples is small. Consequently, many computational methods have been proposed to select genes by using Bayes errors (Zhan & Deng 2007), Random Forest (Diaz & Alvarez 2006), Receiver Operating Characteristics (ROC) (Mamitsuka 2006) or other measures.

In addition to the above gene selection methods, some research efforts (Li et al. 2004; Statnikov et al. 2005) also empirically evaluated different gene selection methods for different types (binary or multi-category) of expression data. In Li et al. (2004), their comparative study concluded that it is difficult to find one best gene selection measure good for all classifiers, and overall, Support Vector Machines (SVM) (Cristianini & Taylor 2000) performs the best. Similar observations were also made in Statnikov et al. (2005) with additional conclusion that ensemble classification does not further improve the best SVM models.

Traditionally, the effectiveness of the gene selection is evaluated through the accuracy of the classifiers built from the selected genes. Given two gene selection methods (A and B) and a user specified number of genes, if a classifier trained from the gene set selected from A is more accurate than the classifier from B, method A is regarded as a better gene selection module. Consequently, the impact of the gene selection with respect to different types of tissue samples is ignored. In practice, many microarray experiments involve imbalanced samples, where one type of examples, *i.e.*, majority class (also called negative class) dominant the whole datasets, whereas other class of examples, *i.e.*, minority class (also called positive class) are rare or extremely rare. For example, in Harvard lung cancer dataset (Kent Ridge), Small Cell Lung Cancer (SCLC) and Squamous cell cancer (SQUA) are only 2.96% and 9.85% of all 203 examples in the dataset. Such data imbalance raises numerous challenges for gene selection research. For example, a gene selection method can be biased and favor majority class examples. In other words, although gene selection may help build a better model in terms of its overall prediction accuracy, such an accuracy improvement may be subjected to a compromise (*i.e.*, a decrease) of the accuracy on minority class examples. Considering that for most, if not all, biological research, the meaning of a predication model is to correctly identify positive examples. A model's prediction accuracy over all examples, without considering the class distribution, might be misleading or even meaningless. Unfortunately, although gene selection has received a lot of attention recently, the impact of gene selection with respect to imbalanced expression data is poorly understood and for which comprehensive empirical studies are lacking.

In this paper, we report our recent study on the impact of gene selection on imbalanced microarray data. We carry out a set of experiments on eleven gene expression datasets by using five types of gene selection methods, and the final results are evaluated based on four supervise learning methods and four empirical measures including overall prediction accuracy, false positive rate, false negative rate, and Kappa statistics. The reminder of the paper is structured as follows. In Section 2, we briefly introduce the experimental settings. Experimental results and analysis are discussed in Section 3. Concluding remarks are reported in Section 4.

2 Experimental Settings

2.1 Benchmark Gene Expression Datasets

Our benchmark dataset consists of 11 gene expression datasets collected from numerous resources (Kent Ridge; Pablo de Olavide), with their data characteristics reported in Table 1. All datasets contain two types of examples, i.e., positive and negative examples (where positive class denotes the minority class). If an original dataset contains multiple classes, we select one particular class (usually a specific cancer type) as the positive class, and treat all remaining classes as the negative class. For example, the Brain and Pancreas datasets are built from the ECML gene expression data (Pablo de Olavide) by selecting Brain tumor and Pancreas tumor as the positive class respectively. By using the above process, we are able to build a set of benchmark datasets with a variety of class imbalance levels, from which we can study the relationship between different types of gene selection measures, learning methods, and different levels of class imbalance.

Table 1. The data characteristics of the benchmark datasets

Name	# of Samples	# of Genes	Pos. vs. Neg.
Brain Tumor	90	27,679	0.26:0.74
Breast Cancer	97	24,481	0.47: 0.53
CentralNerve	60	7,129	0.35:0.65
Colon Cancer	62	2,000	0.35:0.65
DBCL_NIH	240	7,399	0.43:0.57
DBCL_Outcome	58	7,129	0.45:0.55
DBCL_Tumor	77	7,129	0.25:0.75
Pancreas Tumor	90	27,679	0.09:0.91
Lung Cancer	203	12,601	0.32:0.68
Lymphoma	96	4,026	0.24:0.76
Prostate Tumor	136	12,600	0.43:0.57

2.2 Gene Selection Methods

Five commonly used feature selection methods are employed in our study, these include CFS subset (Hall 2000), Chi-Square (χ^2) (Plackett 1983), Information Gain Ratio (IGR) (Quinlan 1993), ReliefF (Robnik & Kononenko 2003), and Symmetrical Uncertainty (SU) (Yu & Liu 2003). All methods are based on their implementation in WEKA data mining tool (Witten & Frank 1999). Among five measures, χ^2 and IGR evaluate correlation between each single gene and the labels of the samples, followed by a ranking process to sort all genes based on their relevance to the learning task. Given gene Y and class label X of all samples, the χ^2 in Eq.(1) is determined by finding the difference between the observed frequency of samples labeled as class X_i , $O_{X_i}^Y$, and the theoretical (expected) frequency E_{X_i} , assume X and Y are independent. Similarly, IGR defined by Eq.s (2) and (3) assess the ratio between the

gain of the entropy of class X, $H(X)$, given gene Y, $H(X|Y)$, and the entropy of considering gene Y only, $H(Y)$. ReliefF assesses the ability of each single gene in differentiating a sample's neighbors from different classes. Given a sample x , ReliefF first finds two sets of neighbors where the first set contains nearest neighbors with the same labels as x , and the second set contains neighbors labeled differently from x . Each single gene is then evaluated based on their ability of differentiating these two sets of examples. For symmetrical uncertainty attribute measure (Yu & Liu 2003), each gene is assessed by the ratio between the information gain of the gene and the sum of the entropy of considering the genes and the sample labels independently, as defined by Eq. (4).

$$\chi^2 = \sum_i \frac{(O_{x_i}^Y - E_{x_i})^2}{E_{x_i}} \tag{1}$$

$$H(X | Y) = H(X) - H(X | Y) \tag{2}$$

$$Gain _ R(X, Y) = \frac{H(X | Y)}{H(Y)} \tag{3}$$

$$SU(X, Y) = 2 \left[\frac{IG(X | Y)}{H(X) + H(Y)} \right] \tag{4}$$

The above four measures (χ^2 , IGR, ReliefF, and SU) consider each single gene independently (we call them ranker measures in this paper). Such methods are known to violate the ground truth where a number of genes usually form a network to inhibit/upregulate the expression of other gene, and thus result in specific diseases. Despite of the above biological realities, it is often suggested that single best features does not necessarily form a best feature subset. Consequently, we employ a feature subset based measure, CFS (Hall 2000), which employs a filter approach (we use genetic search method) to continuously remove a number of genes, until a set of genes remain. In summary, among all five measures, χ^2 , IGR, ReliefF, and SU are single feature based whereas CFS is a subset based measure.

2.3 Learning Methods and Measures

Four learning methods used in our study include Support Vector Machines (SVM), k nearest neighbors (k -NN), Random Forest (RF), and PART which is a C4.5 (Quinlan 1993) based decision rule method. We use default parameter setting in WEKA for all methods except k -NN and RF. For k -NN, we set maximum nearest neighbors to be 30, and use cross-validation approach to find the optimal k values (denoted by C30-NN in this paper). Research suggests that the error rate of a k -NN can approach to Bayes error, *i.e.*, minimum error rate, if proper k value is specified. For Random Forest, we use 100 trees instead of the default setting (10 trees), and use RF-100 to denote RF in this paper. The above four methods represent a variety of learners which were reported to have good performance on gene expression data. For example, SVM and k -NN are two commonly suggested methods for gene expression data (Li et al. 2004; Statnikov et al. 2005). Random Forest is recently reported (Diaz & Alvarez 2006) to

have good performance for gene selection and classification. PART is a decision rule type of method with explicit rules for classification, this is very important for clinical practice which requires transparent decision logics. From this point of view, PART is more informative than SVM, k -NN and RF.

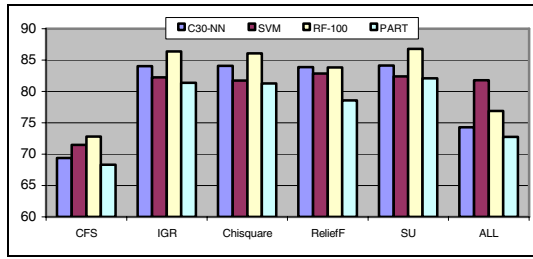
In our study, we use different gene selection measure to select a number of genes (varying from 1, 3, 5, 10, 20, 30, 50, 80, 100, 130, to 150). Existing research suggested (Li et al. 2004) that no further improvement can be observed if the number of selected genes is more than 150. After the selection of the genes, we use data containing the selected genes to build classifiers. All observations are made based on the average results over 10 times 10-fold cross validation, and the measures we employed include overall accuracy (AC), true positive rate (TP), true negative rate (TN), and Kappa statistics (KS). While prediction accuracy reveals the overall performance of different learning methods, TP, TN, and KS show how good the method performs on individual class? So we can conclude that when class distribution in the datasets is imbalanced (biased), how would different methods act on those learning sets?

3 Experimental Results and Analysis

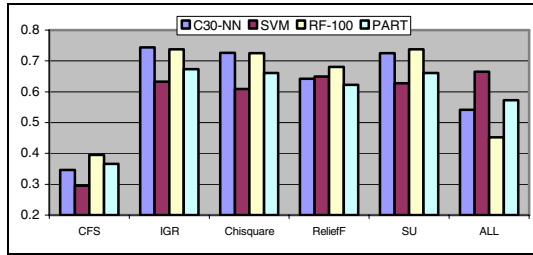
3.1 Average Results over All Benchmark Datasets

The purpose of our first set of experiments is to gain an overall understanding on the performance of different learning methods and different gene selection measures, under the circumstances of gene selection and biased class distribution. For this purpose, we use one measure to select a number of genes (a total of 11 selections with 1, 3, 5, 10, 20, 30, 50, 80, 100, 130, and 150 genes), so each measure generates 11 datasets. After that, we build four classifiers (one for each learning method) from each dataset. We calculate the average results of the classifiers over the 11 gene selections and 11 benchmark datasets, and report there results in Figure 1, where the x -axis denotes different gene selection measures, and the y -axis shows the performance of the classifiers with respect to the selected measure. For comparison purposes, the results from all benchmark datasets without any gene selection are also reported and are denoted by ALL.

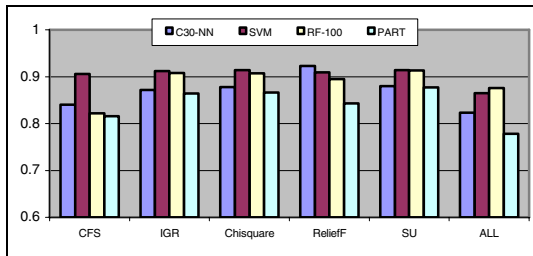
When looking at the average results reported in Figure 1, we can find that the performance of all ranker gene selection measure is relatively close to each other, where the results of IGR and χ^2 are almost identical. Indeed, although IGR and χ^2 use different formulas to characterize each single gene, they both consider the correlation between each gene and the labels of the examples only, and therefore bring almost identical results. On the other hand, the gene subset selection based method (CFS) does not seem to be comparable to ranker measures. The classifiers built from genes selected by CFS are significantly inferior to those selected from ranker measures (this is observed from all four measures AC, TP, TN, and KS). One possible reason is that for CFS, we do not have control on the number of genes the algorithm may output, so we sequentially choose the genes listed on the top of the algorithm's output (which produces around 400 genes). Consequently, this may not produce actual optimal subset from CFS's perspective.



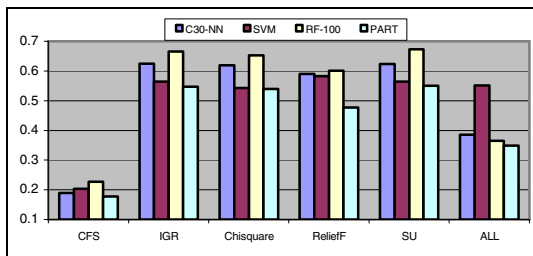
(a) Classification Accuracy (AC)



(b) True Positive Rate (TP)



(c) True Negative Rate (TN)



(d) Kappa statistics (KS)

Fig. 1. The average performance of the classifiers trained from a number of selected genes. In each figure, the bar indicates the performance of each type of classifier with respect to one gene selection measure, and the y-axis indicates the value with respect to the specific measure. Each value is calculated over the average on 11×11 datasets (for each gene expression dataset, we generate 11 datasets, each of which containing a number of selected genes (varying from 1 to 150), and there are 11 benchmark datasets in total)

If we take a look at the results of the classifiers built from the whole dataset (ALL), we can clearly conclude a ranking order SVM, RF-100, C30-NN, and PART. This is consistent with existing study (Li et al. 2004; Statnikov et al. 2005) which concludes that SVM outperforms other learners on high dimensional gene expression data. On the other hand, as long as gene selection is involved, we found that the ranking of the above four learning methods (from the overall accuracy perspective) follows a new order RF-100, C30-NN, SVM, and PART. If we take true positive rate (TP) in Figure 1(b) into consideration, we can find that SVM is actually worse than RF-100, C30-NN, and PART (for IGR, Chi-square, and SU measures). This asserts that SVM can generate accurate unbiased classifiers, if learning is applied to the complete set of genes, however, if a number of genes are selected from the data, a SVM classifier's prediction may be biased to favor majority class (this can be easily confirmed from the results reported in Figure 1(c)). In other words, gene selection has the largest impact on SVM from the positive class perspective.

The Kappa statistic test defines that to which degree a classifier's prediction might be expected by chance. The smaller the kappa value, the more likely a prediction is made by chance. The kappa test results in Figure 1(d) again assert that C30-NN and RF-100 are two methods outperform SVM and PART, if gene selection is applied to the data.

In summary, our major finding in this subsection includes:

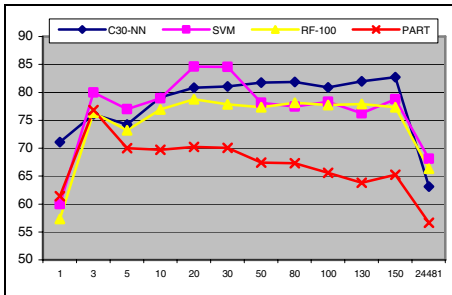
- The results of the ranker gene selection measures are all comparable, and are significantly better than gene subset selection based measures.
- Gene selection can be very beneficial for learners like k -NN, Random Forest, and PART, whereas for SVM, the improvement is not significant or bring negative impact.
- SVM's performance is sensitive to gene selection. A SVM classifier mostly outperforms rival methods if the learning is applied to the whole data without any gene selection, whereas a SVM classifier built from the expression data with reduced number of genes might be severely biased to favor negative class examples.

3.2 Detailed Results on Selected Gene Expression Data

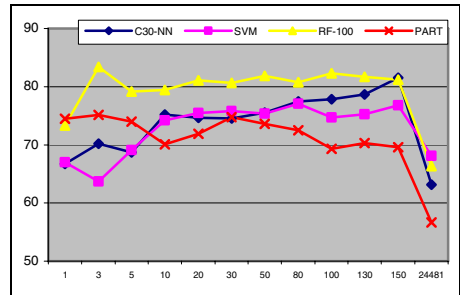
In this subsection, we provide detailed results on imbalanced expression data, where our analysis will focus on individual datasets with a number of selected genes. Because ranker gene selection measures perform comparably, our analysis will focus on IGR and ReliefF (those are also two commonly used feature selection methods in practice). In addition, because our focus is on the impact of gene selection on the imbalanced datasets, we will use three datasets with different levels of imbalance, and report their results on two measures: AC and TP.

Figures 2 and 3 report detailed results on three datasets with different levels of imbalance: Breast (0.47:0.53), Central Nerve Systems (0.35:0.65), and Brain Tumor (0.26:0.74). Empirically, by selecting merely 5 genes, one can mostly build a model better than the one trained from all genes. This applies to all learning methods employed in our study. Meanwhile, Figure 2 also shows that the best models are usually

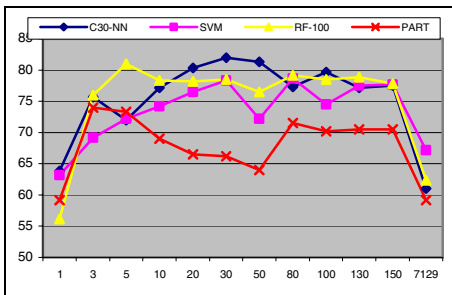
built on a selection of tens of genes only, where most of times, selecting 30 to 50 genes is sufficient to build a good model. Biologically, genes barely function independently but interact with each other as a network. This means that the expression of one gene may trigger other genes to express (or suppress) so cells can naturally adapt to their environments. In practice, the interaction of such a network is confined to a number of genes only (*i.e.*, not all genes interact with each other). Presumably, our results suggest that 30-50 is a reasonable size for genes to form a network in order to fulfill a specific function.



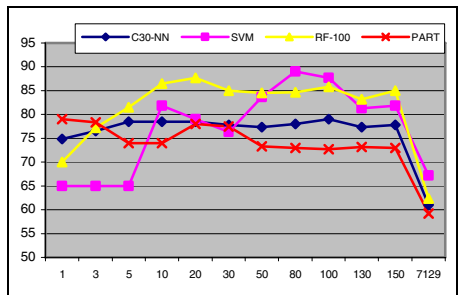
(a.1) Breast Cancer Dataset (ReliefF)



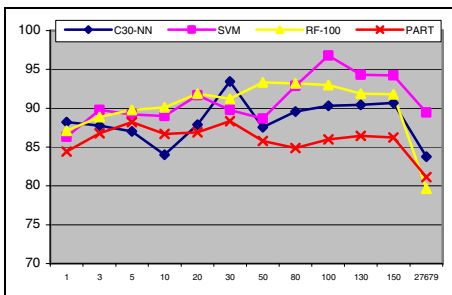
(a.2) Breast Cancer Dataset (IGR)



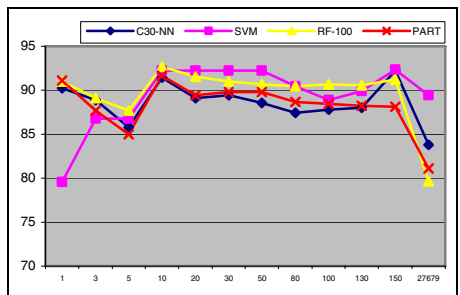
(b.1) Central Nerve Dataset (ReliefF)



(b.2) Central Nerve Dataset (IGR)

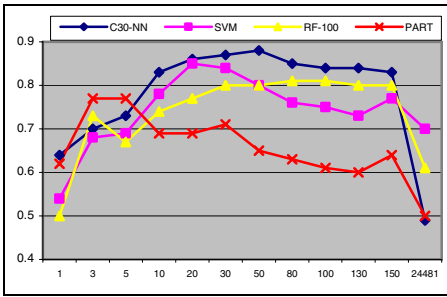


(c.1) Brain Tumor Dataset (ReliefF)

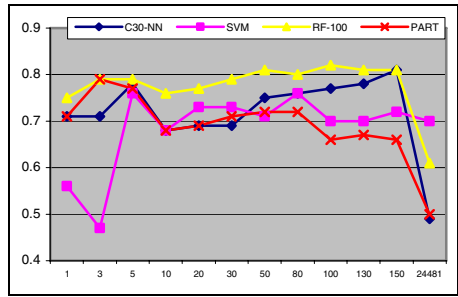


(c.2) Brain Tumor Dataset (IGR)

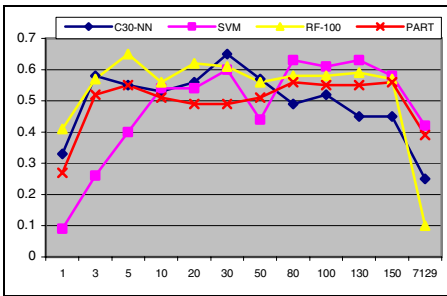
Fig. 2. The overall prediction accuracies of the classifiers trained from different number of selected genes. The x-axis denotes the number of selected genes, and the y-axis indicates the prediction accuracies. The first column shows the results using ReliefF measure, and the second column show the results of the IGR measure.



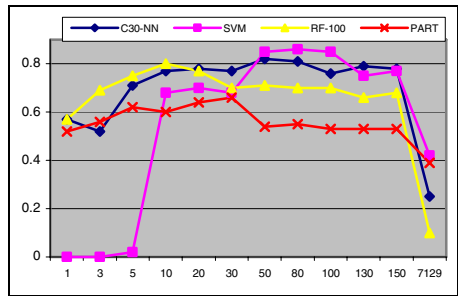
(a.1) Breast Cancer Dataset (ReliefF)



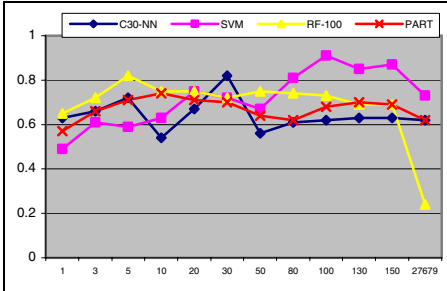
(a.2) Breast Cancer Dataset (IGR)



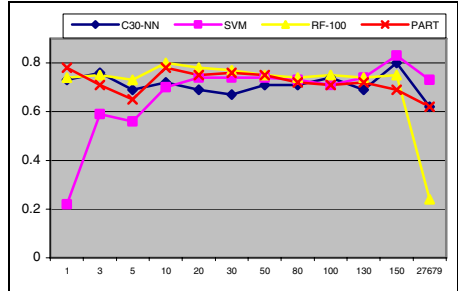
(b.1) Central Nerve Dataset (ReliefF)



(b.2) Central Nerve Dataset (IGR)



(c.1) Brain Tumor Dataset (ReliefF)



(c.2) Brain Tumor Dataset (IGR)

Fig. 3. The true positive rates of the classifiers trained from different number of selected genes. The x -axis denotes the number of selected genes, and the y -axis indicates the true positive rates. The first column shows the results using ReliefF measure, and the second column show the results of the IGR measure.

As shown in Figure 3, when selecting a very small number of genes, say less than 10, SVM classifiers have very poor prediction accuracy on minority class examples. This explains why SVM's TP values reported in Figure 1 are relatively lower than other methods. In addition, when considering the level of imbalance in each datasets, we found that SVM is very sensitive to the information gain measure, where a small number of selected genes almost always result in severely biased SVM classifiers with poor accuracy on the positive examples.

In summary, our major findings in this section include:

- Selecting a very small number, as low as 5, of genes can help build classifiers which outperforms the ones trained from all the genes.
- The best classifiers for gene expression data are usually generated from a dataset with around 50 genes. Under such circumstances, random forest and k -NN are most likely two methods outperforming others, if the number of trees is large and the k value is determined by a cross-validation based approach.
- When the number of selected genes is small, *e.g.*, less than 10, SVM classifiers are severely biased; whereas this bias will gradually disappear once the number of selected genes is reasonably large, say more than 100.

4 Conclusions

Due to the scarcity of tissue samples and many other realities, class imbalance commonly exists in many biological experiments, where the positive examples are significantly fewer than other types of examples. Although numerous researches show that gene selection is beneficial for building accurate prediction models, such conclusions, however, were made without a thorough understanding of the impact of gene selection on imbalanced data. By using 11 benchmark datasets, 5 gene selection measures, 4 learning methods, and 4 measures, the empirically study reported in this paper brought several major findings which have not been revealed in the existing literature: (1) selecting a small number (as less as 5) of genes can help build a model superior to the one from the whole genes; (2) the best classification model is usually built upon on tens of genes only, and adding more genes is likely not helpful to improve the model further; (3) gene selection is mostly beneficial for learners like k -NN, Random Forest, and PART; (4) SVM, a commonly suggested learning method for gene expression data, is extremely sensitive to the number of selected genes, and a small number of genes may result in severely biased SVM classifiers; (5) as long as gene selection is concerned, random forest with a large number of trees and k -NN with cross-validation based k value selection are two promising methods with high prediction accuracy on positive examples, as well as the whole dataset. Based on the above conclusions, we suggest that more research efforts should focus on devising effective measures to select genes which are helpful for positive examples, but not to increase the overall prediction accuracy.

References

- Golub, T., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* 286, 531–537 (1999)
- Xiong, M., et al.: Biomarker identification by feature wrappers. *Genome Research* 11, 1878–1887 (2001)
- Segal, E., et al.: Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics* 34(2), 166–176 (2003)

- Zhan, J., Deng, H.: Gene selection for classification of microarray data based on Bayes error. *BMC Bioinfo.* 8 (2007)
- Diaz, R., Alvarez, S.: Gene selection and classification of microarray data using random forest. *BMC Bioinfo.* 7 (2006)
- Mamitsuka, H.: Selecting features in microarray classification using ROC curves. *Pattern Recognition* 39, 2393–2404 (2006)
- Li, T., et al.: A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics* 20, 2429–2437 (2004)
- Statnikov, A., et al.: A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics* 21(5) (2005)
- Kent Ridge, Kent Ridge Biomedical Data Set Repository, <http://sdmc.i2r.a-star.edu.sg/rp/>
- Witten, Frank, E.: *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco (1999)
- Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlationbased filter solution. In: *Proc. of ICML* (2003)
- Hall, M.A.: Correlation-based feature selection for discrete and numeric class machine learning. In: *Proc. of ICML* (2000)
- Plackett, R.: Karl Pearson and the Chi-Squared Test. *International Statistical Review* 51(1), 59–72 (1983)
- Quinlan, J.: *C4.5: Programs for Machine learning*. M. Kaufmann, San Francisco (1993)
- Robnik, M., Kononenko, I.: Theoretical and empirical analysis of ReliefF and RreliefF. *Machine Learning* 53, 23–69 (2003)
- Cristianini, N., Taylor, J.: *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge (2000)
- Breiman, L.: *Random Forests*. *Machine Learning* (2001)
- Aha, D., Kibler, D., Albert, M.: Instance-based learning algorithms. *Machine learning* 6, 37–66 (1991)
- Pablo de Olavide, Pablo de Olavide University of Seville, Gene Expression Data Repository, <http://www.upo.es/eps/big5/datasets.html>

Spatial Information and Boolean Genetic Regulatory Networks^{*}

Matthieu Manceny^{1,2}, Marc Aiguier¹, Pascale Le Gall^{1,2}, Joan Hérisson²,
Ivan Junier², and François Képès²

¹ École Centrale Paris, MAS Laboratory, France

² Epigenomics Project, University of Évry, France

Abstract. Modelling frameworks for biological networks are used to reason on the models and their properties. One of the main problems with such modelling frameworks is to determine the dynamics of gene regulatory networks (GRN). Recently, it has been observed in *in vivo* experiments and in genomic and transcriptomic studies, that spatial information are useful to better understand both the mechanisms and the dynamics of GRN. In this paper we propose to extend the modelling framework of R. Thomas in order to introduce such spatial information between genes, and we will show how these further informations allow us to restrict the number of dynamics to consider.

Keywords: Genetic Regulatory Networks, Spatial Information, Boolean Dynamics, Discrete Mathematical Modelling.

1 Introduction

To understand Genetic Regulatory Networks (GRN), modelling frameworks and simulation techniques are often useful since the complexity of the interactions between constituents of the network (mainly genes and proteins) makes intuitive reasoning difficult. Most of the time, parameters of the model have to be inferred from a set of biological experiments. Formal methods, such as model checking or symbolic execution ([12]), have been proved useful to determine values of parameters leading to valid dynamics of GRN, that is dynamics consistent with biological properties expressed using temporal logic. Nevertheless, these techniques are in practice difficult to manage because biological systems are either large, complex or incompletely known, resulting in a huge number of parameters to consider. Hence, in order to reduce this number, it seems relevant to embed within the model some biological knowledge such as spatial relation between genes.

Recent experiments have shown that both in eukaryotes [3] and in bacteria [4] gene transcription occurs in discrete foci where several RNA polymerases (the transcribing elements) are co-localized. This suggests that genes also tend to co-localize in space in order to optimize transcription rates. Such a scenario is

^{*} This work is performed within the European project GENNETEC (STREP 34952).

supported by genomic and transcriptomic analysis [5,6]. These have revealed that the genes which are regulated by a given transcription factor and the gene which codes for the transcription factor tend to be located periodically along the DNA [5]. In this way, the genes can be easily co-localized in the three-dimensional space according to a solenoidal structure of the DNA/chromatin, even in the presence of several kinds of transcription factors [7]. As a result, the effect of a transcription factor is enhanced due to the spatial proximity of the targets. This phenomenon is reminiscent of the local concentration effect that has been uncovered by Müller-Hill [8] a decade ago. Local concentration simply means that the interaction between molecules that are able to interact with each other is all the more efficient when molecules are close to each other. This straightforward statement is crucial to understand genome organization because genomes seem to have evolved in order to optimize the spatial proximity of reactive groups [7,8,9].

In this article, we propose to include spatial information into GRN and to study its effect upon the dynamics of the network. Our approach is based on the discrete modelling of GRN that has been introduced by René Thomas [10]. The spatial information concerns the gene proximity that results from a specific organization of DNA/chromatin. This proximity is modelled through the notion of privileged interaction between genes which is an ubiquitous concept in biology. For instance, specific interactions (e.g. between a transcription factor and DNA) in contrast to non-specific interactions, or local concentration phenomena are examples of privileged interactions. The use of privileged interaction is mainly based on the idea that if two interactions lead to contradictory effects, then the privileged interaction is preferred to the non privileged one.

The paper is structured as follows. Section 2 presents our model of GRN including privileged interactions. In Section 3, we are interested in the Boolean dynamics of such GRN. The dynamics is governed by a set of so called logical parameters, and we present how the structure of the GRN determines the possible values of these parameters. Nevertheless, the possible dynamics still remain too numerous, and so, Section 4 presents how to use privileged interactions to reduce the number of dynamics to consider. Section 5 presents a illustrative example, and some numerical simulations. Finally, Section 6 gives some concluding remarks.

2 GRN with Privileged Interactions (PGRN)

Genetic Regulatory Networks are usually represented by an oriented graph, called *interaction graph*, whose nodes abstract the proteins or genes which play a role in the system and edges abstract the known interactions of the GRN. The model of this article is based on Boolean GRN, that is GRN where gene can only have two *expression levels* (see Section 3). An interaction ($a \rightarrow b$) can be either an activation or an inhibition: in an *activation*, the increase of the expression level of a leads to an increase of the expression level of b , the edge is labelled by the sign $+$ and a is an activator of b ; in an *inhibition*, the increase of a leads

to a decrease of b , the edge is labelled by the sign $-$ and a is an inhibitor of b . To this classic representation, we add the notion of *privileged interactions* as a subset of the interactions of the GRN.

Definition 1 (PGRN: GRN with privileged interactions). A GRN with privileged interactions (*PGRN*) is a labelled directed graph $G = (V, E, S, P)$ where (V, E, S) is an interaction graph that is V is a finite set whose elements are called variables, $E \subseteq V \times V$ is the set of interactions, and $S : E \rightarrow \{+, -\}$ associates to each interaction its sign (" $+$ " for activation and " $-$ " for inhibition); and $P \subseteq E$ is the set of privileged interactions.

For any $i \in V$, $V^-(i)$ (resp. $V^+(i)$) denotes the set of predecessors (resp. successors) of i , that is elements of V which have an action on i (resp. on which i has an action): $V^-(i) = \{j | j \in V, (j, i) \in E\}$, $V^+(i) = \{j | j \in V, (i, j) \in E\}$; $P(i)$ denotes the set of privileged predecessors of i : $P(i) = \{j | j \in V^-(i), (j, i) \in P\}$.

Definition 2 (Activators and inhibitors). Let (V, E, S, P) be a PGRN, and let $i \in V$ be a gene. We denote by $A(i)$ (resp. $I(i)$) the set of activators (resp. inhibitors) of i : $A(i) = \{j | j \in V^-(i), S(j, i) = +\}$ and $I(i) = \{j | j \in V^-(i), S(j, i) = -\}$.

In the following, a PGRN will be represented as a graph where nodes are variables, arrows are interactions (dashed arrows for the privileged ones) and signs label arrows (see Fig. 3).

Example 1 (Example of interaction graph). Let us exemplify Definition 1 with the toy interaction graph (that is without any information on privileged interactions) from Fig. 1 where a gene i is inhibited by j_1 and j_2 and activated by k . Section 3 will present the dynamics of such a graph; the influence of privileged interactions among these three interactions is presented in Section 4.

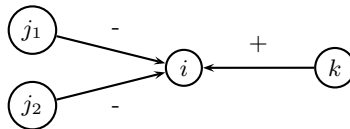


Fig. 1. Example of interaction graph

3 Boolean Dynamics of PGRN

3.1 Boolean Dynamics and Logical Parameters

In *Boolean dynamics*, genes can attain two levels, called *expression levels*: *effective* denoted by 1, or *ineffective* denoted by 0. The knowledge of the expression levels of all the genes define a *Boolean dynamic state*.

Definition 3 (Boolean dynamic states). Let $G = (V, E, S, P)$ be a PGRN, and let $i \in V$ be a gene. We denote by $\mathbb{X}(G)$ the set of Boolean dynamic states of G : $\mathbb{X}(G) = \{0, 1\}^{|V|}$. For $x = (x_1, \dots, x_{|V|}) \in \mathbb{X}(G)$, $x_i \in \{0, 1\}$ is the expression level of gene i in x .

The dynamics of a PGRN consists in the evolution of each gene’s expression level step by step. This evolution for a given gene does not depend on all the genes of the PGRN, but only on the genes which have an action on the given gene, that is its *effective predecessors*.

Definition 4 (Effective predecessors). Let $G = (V, E, S, P)$ be a PGRN, and let $i \in V$ be a gene. Let $x \in \mathbb{X}(G)$ be a dynamic state. We denote by $A^*(i, x)$ (resp. $I^*(i, x)$, $w^*(i, x)$) the set of effective activators (resp. effective inhibitors, effective predecessors) of i in the state x : $A^*(i, x) = \{j | j \in V^-(i), S(j, i) = +, x_j = 1\}$, $I^*(i, x) = \{j | j \in V^-(i), S(j, i) = -, x_j = 1\}$ and $w^*(i, x) = A^*(i, x) \cup I^*(i, x)$.

Several dynamics can be associated to a given PGRN. These dynamics are described by a set of *logical parameters* which associates the future expression level of a given gene according to its effective predecessors.

Definition 5 (Logical parameters). Let (V, E, S, P) be a PGRN. For $i \in V$, we denote by $K_i : 2^{V^-(i)} \rightarrow \{0, 1\}$ the set of logical parameters associated to i .

Example 2 (Logical parameters). In Fig. 1, gene i has three predecessors. Thus, there is 8 logical parameters K_i to consider: $K_i(\emptyset)$, $K_i(\{j_1\})$, $K_i(\{j_2\})$, $K_i(\{k\})$, $K_i(\{j_1, j_2\})$, $K_i(\{j_1, k\})$, $K_i(\{j_2, k\})$ and $K_i(\{j_1, j_2, k\})$.

For example, the logical parameter $K_i(\{j_2, k\})$ represents i ’s next expression level when the dynamic state is such that $x_{j_1} = 0$, $x_{j_2} = 1$ and $x_k = 1$.

Determining the dynamics of a PGRN consists in the attribution of values to the different logical parameters. The number of the possible attributions is huge: given a gene i , there are $2^{|V^-(i)|}$ logical parameters K_i , and each parameter can take two values. Thus, we have to consider $\prod_{i \in V} 2^{|V^-(i)|}$ possible attributions. For example, just for the interaction graph from Fig. 1 we have to consider $2^{2^3} = 256$ possibilities. Nevertheless, the structure of the interaction graph restricts the possible values of logical parameters.

3.2 Valid Logical Parameters

The values of logical parameters of an interaction graph must satisfy some constraints, linked to the graph structure and to the type of interaction. Logical parameters respecting the following constraints are said to be *valid*.

The *Definition constraint* is based on the definition of activation and inhibition. If a gene j which activates a gene i becomes effective, then we cannot be sure that i becomes itself effective (it may be inhibited by other genes), but the expression level of i cannot decrease.

¹ Let us recall that $|V|$ denotes the number of elements in the set V .

Constraint 1 (Definition). Let (V, E, S, P) be a PGRN, and let i, j in V be two genes such that $j \in V^-(i)$. If $S(j, i) = +$ then $\forall \omega \subseteq V^-(i), K_i(\omega) \leq K_i(\omega \cup \{j\})$. If $S(j, i) = -$ then $\forall \omega \subseteq V^-(i), K_i(\omega) \geq K_i(\omega \cup \{j\})$.

The *Observation constraint* expresses how we identify that a predecessor is an activator or an inhibitor. If j is an activator of i , then it exists at least one dynamic state where the effectiveness of j leads to an increase of the expression level of i . In other word, at least one of the previous inequalities is strict.

Constraint 2 (Observation). Let (V, E, S, P) be a PGRN, and let i, j in V be two genes such that $j \in V^-(i)$. If $S(j, i) = +$ then $\exists \omega \subseteq V^-(i), K_i(\omega) < K_i(\omega \cup \{j\})$. If $S(j, i) = -$ then $\exists \omega \subseteq V^-(i), K_i(\omega) > K_i(\omega \cup \{j\})$.

Finally, the *Maximum constraint* expresses that in a dynamic state where all the activators of a gene are effective and simultaneously none of the inhibitors is effective, then the gene is effective. Conversely, if none of the activators is effective, and all inhibitors are, then the logical parameter is equal to 0.

Constraint 3 (Maximum). Let (V, E, S, P) be a PGRN, and let i in V be a gene. Then: $K_i(A(i)) = 1$, and $K_i(I(i)) = 0$.

Example 3 (Valid parameters). Let us consider the interaction graph from Fig. 1. The Maximum constraint imposes that $K_i(\{k\}) = 1$ and $K_i(\{j_1, j_2\}) = 0$. Other relations between parameters are resumed in Fig. 2, where an arrow from a node K to a node K' means $K \geq K'$ (Definition constraint), and this inequality is strict (Observation constraint) for at least one arrow of each type (plain, dashed or dotted arrows). All three constraints taking into account, there are only 9 valid sets of parameters.

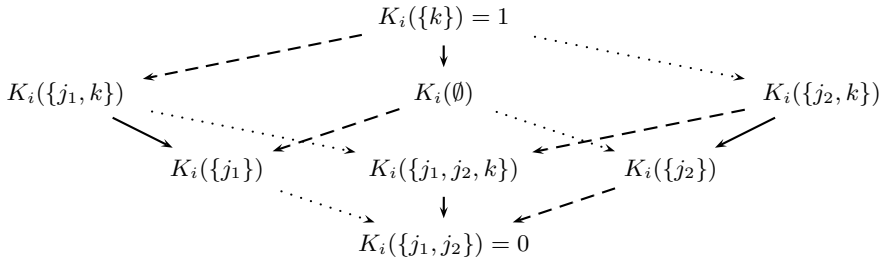


Fig. 2. Relation among logical parameters of the interaction graph from Fig. 1

4 Toward a Reduction of Valid Dynamics

4.1 Conflicts and Dilemma

Despite the above constraints, valid dynamics of PGRN still remain too numerous. The different dynamics exist due to some dynamics states where the three constraints do not allow us to determine unique values for logical parameters: *Conflicts* occur when a gene is simultaneously activated and inhibited, *Dilemma* occur when all the activators (resp. inhibitors) of a gene are not effective.

Definition 6 (Conflicts and dilemma). Let $G = (V, E, S, P)$ be an interaction graph, let $i \in V$ be a gene and let $x \in \mathbb{X}(G)$ be a dynamic state. x is a situation of conflict for gene i iff $A^*(i, x) \neq \emptyset$ and $I^*(i, x) \neq \emptyset$. x is a situation of dilemma for gene i iff $(A^*(i, x) \neq \emptyset$ and $A^*(i, x) \neq A(i))$ or $(I^*(i, x) \neq \emptyset$ and $I^*(i, x) \neq I(i))$

In the following, we will focus on the determination of logical parameters. Thus, conflicts and dilemma will refer to parameters, that is $K_i(w^*(i, x))$ is a conflict (resp. a dilemma) if and only if x is a situation of conflict (resp. dilemma) for gene i . In other words, if $w^*(i, x) = \omega$, then $K_i(\omega)$ is a conflict iff $\omega \cap A(i) \neq \emptyset$ and $\omega \cap I(i) \neq \emptyset$; $K_i(\omega)$ is a dilemma iff $A(i) \not\subseteq \omega \not\subseteq I(i)$ or $I(i) \not\subseteq \omega \not\subseteq A(i)$.

Note that, in this model, $K_i(\emptyset)$ is neither a conflict nor a dilemma, but corresponds to the basal situation, where a gene i is not activated or inhibited.

Example 4 (Conflicts and dilemma). Let us consider the 8 possible dynamic states and the associated logical parameters for gene i for the interaction graph from fig. 1. $K_i(\{j_1\})$ and $K_i(\{j_2\})$ are dilemma; $K_i(\{j_1, j_2, k\})$ is a conflict; $K_i(\{j_1, k\})$, $K_i(\{j_2, k\})$ are both conflicts and dilemma. $K_i(\{k\})$ and $K_i(\{j_1, j_2\})$ are neither conflict nor dilemma: the former correspond to a situation where i is fully activated and is not inhibited, the latter corresponds to the reverse situation.

4.2 Constraints Based on Privileged Interactions

By definition, privileged interactions are such that their force is higher than the force of non privileged interactions. Figure 3 illustrates how to solve conflicts and dilemma using the privileged interactions: for conflicts, if two interactions occur simultaneously, then the privileged one is preferred; a dilemma is solved if one of the present gene is a privileged one.

This idea is captured through two constraints on logical parameters. The first constraint, called *Direct influence* indicates that if none of privileged activators (resp. inhibitors) is effective, and some privileged inhibitors (resp. activators) of the considered gene are effective, then the expression level is 0 (resp. 1).

Constraint 4 (Direct influence). Let $G = (V, E, S, P)$ be a PGRN. Let $i \in V$ be a gene and $x \in \mathbb{X}(G)$ be a Boolean dynamic state. If $A^*(i, x) \cap P(i) \neq \emptyset$ and $I^*(i, x) \cap P(i) = \emptyset$ then $K_i(w^*(i, x)) = 1$. If $I^*(i, x) \cap P(i) \neq \emptyset$ and $A^*(i, x) \cap P(i) = \emptyset$ then $K_i(w^*(i, x)) = 0$.

The second constraint, called *Relative influence*, states that expression levels of non privileged predecessors is not important compared to the presence or absence of privileged ones. In other words, the value of a logical parameter for a set of effective genes, whose at least one is a privileged predecessor, remains the same whatever non privileged predecessors becoming effective.

Constraint 5 (Relative influence). Let (V, E, S, P) be a PGRN. Let $i \in V$ be a gene and let $\omega \subseteq V^-(i)$ be a set of predecessors of i such that $\omega \cap P(i) \neq \emptyset$. Let $j \in V^-(i)$ be a gene such that $j \notin P(i)$. Then: $K_i(\omega \cup \{j\}) = K_i(\omega)$.

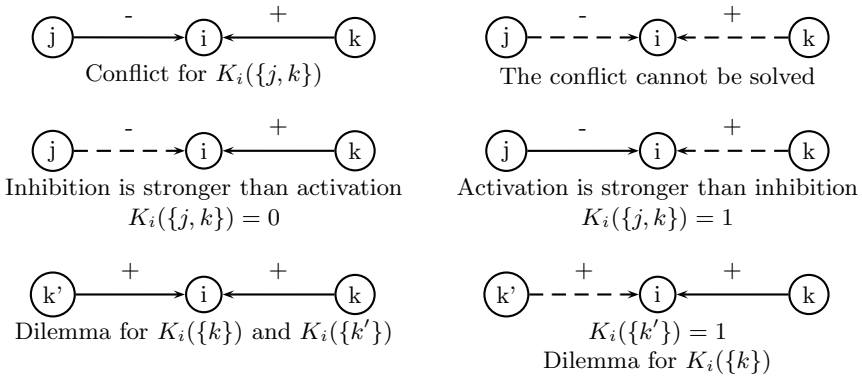


Fig. 3. Solving conflicts and dilemma with privileged interactions

Example 5 (Influence of privileged interactions). Let us suppose that j_1 is the only privileged predecessor in Fig. 1. Then, as soon as j_1 is ineffective, conflict and dilemma appears between other genes, but when j_1 is effective, they are solved. The 9 valid sets of parameters are reduced to 2. If we now suppose that k is the only privileged predecessor, there is no conflict, but some dilemma remains, which reduced the number of dynamics to consider to 2. If j_1 and k are privileged predecessors, there are still conflict and dilemma, but the number of dynamics to consider is reduced to 2. Finally, if we suppose that both j_1 and j_2 are privileged predecessors, then there is neither conflict nor dilemma, and the dynamics is unique.

4.3 Unique Dynamics

We present here conditions to obtain, given a PGRN, a unique set of parameters leading to a unique dynamics. Obviously, if some genes have no predecessor, we cannot determine their expression levels, which in fact do not evolve along the time. A necessary and sufficient condition to have *no conflict* is that the set of privileged predecessors is either equal to activators or inhibitors.

Theorem 1 (No conflict). *The conflict situations of a PGRN (V, E, S, P) can be solved iff for all $i \in V$, $P(i) = A(i)$ or $P(i) = I(i)$*

Proof. Sufficient. Let x be a situation of conflict for gene i : $A^*(i, x) \neq \emptyset$ and $I^*(i, x) \neq \emptyset$. Let us suppose that $P(i) = A(i)$ (the proof is similar for $P(i) = I(i)$). Then we have $I^*(i, x) \cap P(i) = \emptyset$ and $A^*(i, x) \cap P(i) = A^*(i, x)$. Thus, due to the constraint of direct influence, $K_i(w^*(i, x)) = 1$ and the conflict is solved.

Necessary. Let us suppose that the condition is not verified for a given gene i , that is $P(i) \neq A(i)$ and $P(i) \neq I(i)$. $P(i) \neq A(i)$ iff either it exists $k \in A(i) \setminus P(i)$ or it exists $j \in I(i) \cap P(i)$; $P(i) \neq I(i)$ iff either it exists $j' \in I(i) \setminus P(i)$ or it exists $k' \in A(i) \cap P(i)$. If it exists $k \in A(i) \setminus P(i)$ and it exists $j' \in I(i) \setminus P(i)$, then the situation x where the only effective genes are k and j' is a situation of conflict. If it exists $k \in A(i) \setminus P(i)$ and it exists $k' \in A(i) \cap P(i)$, then two cases

must be considered: if $I(i) \cap P(i) = \emptyset$ then, with $j'' \in I(i)$, the situation x where the only effective genes are k and j'' is a situation of conflict; if $I(i) \cap P(i) \neq \emptyset$ then, with $j'' \in I(i) \cap P(i)$, the situation x where the only effective genes are k' and j'' is a situation of conflict.

Nevertheless, if all privileged predecessors are ineffective, then a situation of dilemma may occur. Dilemmas occur when two genes having the same action (either activation or inhibition) are not effective simultaneously. Thus, a necessary and sufficient condition to have *no dilemma* is that either there is only one gene for a given action, or each predecessor having this type of action is a privileged predecessor of the target.

Theorem 2 (No dilemma). *The dilemma situations of PGRN (V, E, S, P) can be solved iff for all $i \in V$, $(A(i) \subseteq P(i) \text{ or } |A(i)| = 1)$ and $(I(i) \subseteq P(i) \text{ or } |I(i)| = 1)$.*

Proof. Sufficient. Let us consider the case of activation (the proof is similar for inhibition). Obviously, if $|A(i)| = 1$, then there is no dilemma. If $A(i) \subseteq P(i)$, then: for all $\omega \subseteq A(i)$, if $\omega \neq \emptyset$ then $K_i(\omega) = 1$ due to the constraint of direct influence; for all $\omega_a \subseteq A(i)$, for all $\omega_i \subseteq I(i) \setminus P(i)$, if $\omega_a \neq \emptyset$ then $K_i(\omega_a \cup \omega_i) = 1$, due to the constraint of relative influence; the remaining cases correspond to situations of conflict where both activators and predecessors are privileged predecessors of i .

Necessary. Let us suppose that the condition is not verified. Let us suppose we have $|A(i)| > 1$ and $A(i) \not\subseteq P(i)$ (the proof is similar for the inhibition). Then it exists $a \in A(i) \setminus P(i)$, and the situation x where a is the only effective predecessor of i is a situation of dilemma.

Theorem 3 (No conflict nor dilemma). *Conflict and dilemma situations of a PGRN (V, E, S, P) can be solved iff for all $i \in V$, $(A(i) = P(i) \text{ and } |I(i)| = 1)$ or $(|A(i)| = 1 \text{ and } I(i) = P(i))$*

Proof. The theorem is a direct consequence of theorems 1 and 2.

Under the conditions of this theorem, only one dynamics is consistent with all constraints. Obviously, these conditions are difficult to state in practice. Section 5 will nevertheless illustrate that in any case, the consideration of privileged interactions allows us to reduce the set of consistent dynamics.

5 Influence of Privileged Interactions on Dynamics

5.1 From a Biological Case Study

Pseudomonas aeruginosa are bacteria that secrete mucus (alginate) in lungs affected by cystic fibrosis, but not in common environment. As this mucus increases respiratory deficiency, this phenomenon is a major cause of mortality. Details of the regulatory network associated with the mucus production by *Pseudomas*

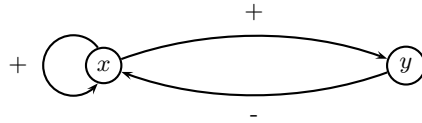


Fig. 4. Interaction graph for the mucus production system in *P. aeruginosa*

aeruginosa are described by Govan and Deretic [11] but a simplified genetic regulatory network has been proposed by Guespin and Kaufman [12], see Fig. 4.

It has been observed that mucoid *P. aeruginosa* can continue to produce mucus isolated from infected lungs. It is commonly thought that the mucoid state of *P. aeruginosa* is due to a mutation which cancels the inhibition of gene x . An alternative hypothesis has been made: this mucoid state can occur by reason of an epigenetic modification, *i.e.* without mutation [12]. The models compatible with this hypothesis are constructed in [1].

The logical parameters to consider are $K_y(\emptyset)$ and $K_y(\{x\})$ for the gene y and $K_x(\emptyset)$, $K_x(\{x\})$, $K_x(\{y\})$ and $K_x(\{x, y\})$ for gene x , which leads without further consideration, to $2^2 \times 2^4 = 64$ possible dynamics. Obviously, this number is decreased considering the constraints previously presented. $K_y(\emptyset) = 0$ and $K_y(\{x\}) = 1$ due to the observation rule. The maximum rule leads to $K_x(\{x\}) = 1$ and $K_x(\{y\}) = 0$, and then the observation rule leads to two possible dynamics: either $(K_x(\emptyset) = 1$ and $K_x(\{x, y\}) = 1)$ or $(K_x(\emptyset) = 0$ and $K_x(\{x, y\}) = 0)$.

The two possible dynamics are due to the conflict between x and y , and then the knowledge of privileged interactions among the activation of x by itself or the inhibition of x by y would lead to the determination of a unique dynamics. If both the interactions are privileged ones (or conversely are not privileged ones) then the two dynamics remain valid. If the inhibition is privileged and not the activation, then $K_x(\emptyset) = 0$ and $K_x(\{x, y\}) = 0$. If the activation is privileged and not the inhibition, then $K_x(\emptyset) = 1$ and $K_x(\{x, y\}) = 1$.

5.2 From Artificial PGRN

In order to estimate the reduction in number of models induced by the introduction of privileged interactions, we have randomly generated PGRN. The generation is parameterized by three values: n the number of genes, p the number of predecessors of a gene and r a ratio to determine which interactions are privileged. We first generate n genes; for each gene we then randomly select p predecessors among the n genes, each one being a privileged predecessor with a probability r . Fig. 5 presents some results on artificial PGRN composed of $n = 10, 25, 50$ and 100 genes. We give one table by hypothesis on the considered number of predecessors: the first three tables correspond to situations where each gene has exactly $p = 2, 3$ or 4 predecessors, and the last table to a situation where each gene has a random number of predecessors between 1 and 4. We chose these rather small values for the number of predecessors per gene to fit a realistic ratio between number of genes and number of interactions.

For each PGRN we evaluate the number of dynamics without any constraint (row named "Total" in each table). We then compute the number of dynamics when all the constraints (definition, observation, maximum, direct and relative

Privileged ratio r	Number of genes n			
	10	25	50	100
0	1024	$3 \cdot 10^7$	$1 \cdot 10^{15}$	$1 \cdot 10^{30}$
1/10	408	$2 \cdot 10^7$	$7 \cdot 10^{12}$	$6 \cdot 10^{25}$
1/5	174	$2 \cdot 10^5$	$5 \cdot 10^{11}$	$3 \cdot 10^{21}$
1/2	22	1171	$1 \cdot 10^6$	$9 \cdot 10^{11}$
1	17	1493	$1 \cdot 10^6$	$6 \cdot 10^{12}$
<i>Total</i>	10^{12}	10^{30}	10^{60}	$2 \cdot 10^{120}$

Each gene has $p = 2$ predecessors

Privileged ratio r	Number of genes n			
	10	25	50	100
0	$3 \cdot 10^9$	$7 \cdot 10^{23}$	$5 \cdot 10^{47}$	$2 \cdot 10^{95}$
1/10	$4 \cdot 10^8$	$6 \cdot 10^{20}$	$1 \cdot 10^{41}$	$2 \cdot 10^{83}$
1/5	$2 \cdot 10^7$	$2 \cdot 10^{18}$	$4 \cdot 10^{35}$	$6 \cdot 10^{67}$
1/2	$4 \cdot 10^4$	$1 \cdot 10^{11}$	$2 \cdot 10^{20}$	$2 \cdot 10^{38}$
1	$3 \cdot 10^5$	$1 \cdot 10^{13}$	$6 \cdot 10^{26}$	$6 \cdot 10^{47}$
<i>Total</i>	$1 \cdot 10^{24}$	$1 \cdot 10^{60}$	$2 \cdot 10^{120}$	$6 \cdot 10^{240}$

Each gene has $p = 3$ predecessors

Privileged ratio r	Number of genes n			
	10	25	50	100
0	$3 \cdot 10^{20}$	$2 \cdot 10^{51}$	$7 \cdot 10^{102}$	—
1/10	$6 \cdot 10^{18}$	$2 \cdot 10^{46}$	$7 \cdot 10^{88}$	—
1/5	$3 \cdot 10^{16}$	$9 \cdot 10^{41}$	$1 \cdot 10^{75}$	—
1/2	$2 \cdot 10^9$	$2 \cdot 10^{21}$	$3 \cdot 10^{38}$	—
1	$1 \cdot 10^{14}$	$6 \cdot 10^{33}$	$4 \cdot 10^{61}$	—
<i>Total</i>	$1 \cdot 10^{48}$	$2 \cdot 10^{120}$	$6 \cdot 10^{240}$	$4 \cdot 10^{481}$

Each gene has $p = 4$ predecessors

Privileged ratio r	Number of genes n			
	10	25	50	100
0	$2 \cdot 10^{12}$	$2 \cdot 10^{29}$	$1 \cdot 10^{54}$	$2 \cdot 10^{101}$
1/10	$7 \cdot 10^{11}$	$5 \cdot 10^{24}$	$3 \cdot 10^{41}$	$6 \cdot 10^{83}$
1/5	$3 \cdot 10^8$	$2 \cdot 10^{21}$	$1 \cdot 10^{38}$	$8 \cdot 10^{63}$
1/2	$2 \cdot 10^4$	$1 \cdot 10^{10}$	$1 \cdot 10^{18}$	$7 \cdot 10^{36}$
1	$1 \cdot 10^7$	$3 \cdot 10^{13}$	$1 \cdot 10^{25}$	$1 \cdot 10^{48}$
<i>Total</i>	$1 \cdot 10^{33}$	$1 \cdot 10^{73}$	$1 \cdot 10^{140}$	$1 \cdot 10^{265}$

Each gene has between 1 and 4 predecessors

Fig. 5. Number of Dynamics for Artificial PGRN

influence) are applied, for several ratios of privileged interactions: when there is no privileged interaction (row "0"), when one interaction out of ten is privileged (row "1/10"), one out of five (row "1/5"), one out of two (row "1/2") and when all interactions are privileged ones (row "1"). Let us note that results between row "1" and row "0" may be largely different, since when all predecessors are privileged (row "1"), then the effectiveness of only one of them allows us to solve dilemma unsolved in row "0". All the values in the different tables given in Fig. 5 are the result of an arithmetic mean over 100 tests. The column "100 genes" for the hypothesis "4 predecessors per gene" is left empty, due to the excessive required computation time.

Obviously, the number of dynamics we have to deal with is huge (at least 10^{12} , see row "Total"), and this number is squared when the number of genes doubles, or when the number of predecessors is increased by one. When considering the constraints of definition, observation and maximum, the number of dynamics is already significantly reduced (see row "0" where none of the interactions is privileged). With the constraints induced by the introduction of privileged interactions (direct and relative influence), the number of dynamics still decreases and the best results are obtained when half of interactions are privileged ones (row "1/2"). Nevertheless, let us point out that the improvement is clearly observed even with small information. For example, when only one interaction out of ten is privileged (row "1/10"). we can observe that in the fourth table, the number of dynamics is divided by 10 for a ten genes network, by 10^5 for 25 genes, and by 10^{18} for 100 genes.

These few simulations illustrate that as soon as spatial information is known, the set of all possible dynamics is really restricted. To go further in this restriction, one can express temporal properties to characterise some knowledge about the behaviour of the GRN. Formal techniques, most of them based on model checking [1], have been applied to select valid dynamics, that is dynamics consistent with biological experiments expressed by temporal properties. The problem is that these formal techniques rapidly become intractable because dynamics associated to the GRN are most of the time very numerous. Thus, from a general point of view, the set of GRN dynamics is all the more reduced than all biological knowledge, including spatial information, is taken into account.

6 Concluding Remarks

In this article we have presented a simple way to include spatial information within the René Thomas' framework of GRN. This supplementary information is described as a property of interactions: an interaction is privileged when the source and target genes are known to be spatially close. In the framework of Boolean dynamics, values of logical parameters are weakly constrained, leading to situations of conflicts or dilemmas where several dynamics are possible. With the notion of privileged interactions, we have determined conditions to solve some of these situations.

The spatial oriented framework we have defined is based on René Thomas' Boolean dynamics and presents the two following advantages. Firstly, since the dynamics for our spatial framework are chosen among classical René Thomas' Boolean dynamics associated to the underlying GRN without privileged interaction, then our dynamics are clearly included in the usual dynamics of GRN. Secondly, since spatial information allows us to solve some conflicts and dilemmas, and thus to determine some logical parameters, the number of dynamics is in practice considerably reduced.

In the goal of validating our approach, we are facing to the fact that, although spatial information seems to be central in order to apprehend the complexity of biological networks, experimental data are rare. Indeed, available data mainly concern large GRN, which are for the moment hardly attainable with our approach due to the high number of parameters to consider. Nevertheless our approach seems particularly adapted, since the first results appear even with few information on spatial relation.

An extension of this work we are particularly interested in deals with multivalued dynamics. In such framework, expression levels of genes are not Boolean, but can take a finite number of values. To each interaction is associated a *threshold* which correspond to the expression level the source gene must exceed in order to the interaction to become effective. Thus, given an interaction graph, the number of dynamics to consider is even higher than in Boolean dynamics. In such a context, the spatial information to be considered will be composed of privileged interactions as in the Boolean case, but also of the notion of *cluster* which expresses co-regulation. Co-regulated genes are spatially close genes

expressed at the same time due to the expression of a single regulating gene. Thus, interactions regulating a cluster are labelled by the same threshold value and this represents a new factor of reduction of the set of multivalued dynamics.

References

1. Bernot, G., Comet, J.P., Richard, A., Guespin, J.: Application of formal methods to biological regulatory networks: Extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology* 229(3), 339–347 (2004)
2. Mateus, D., Gallois, J.P., Comet, J.P., Le Gall, P.: Symbolic modeling of genetic regulatory networks. *Journal of Bioinformatics and Computational Biology* (2007)
3. Jackson, D.A., Hassan, A.B., Errington, R.J., Cook, P.R.: Visualization of focal sites of transcription within human nuclei. *J. Cell Biol.* 164, 515–526 (2004)
4. Cabrera, J.E., Jin, D.J.: The distribution of rna polymerase in *escherichia coli* is dynamic and sensitive to environmental cues. *Mol. Microbiol.* 50(5), 1493–1505 (2003)
5. Képès, F.: Periodic transcriptional organization of the *e.coli* genome. *J. Mol. Biol.* 340(5), 957–964 (2004)
6. Carpentier, A.S., Torresani, B., Grossmann, A., Henaut, A.: Decoding the nucleoid organisation of *bacillus subtilis* and *escherichia coli* through gene expression data. *BMC Genomics* 6(1), 84 (2005)
7. Képès, F., Vaillant, C.: Transcription-based solenoidal model of chromosomes. *Complexus* 1(4), 171–180 (2003)
8. Muller-Hill, B.: The function of auxiliary operators. *Mol. Microbiol.* 29(1), 13–18 (1998)
9. Lanctot, C., Cheutin, T., Cremer, M., Cavalli, G., Cremer, T.: Dynamic genome architecture in the nuclear space: regulation of gene expression in three dimensions. *Nat. Rev. Genet.* 8(2), 104–115 (2007)
10. Thomas, R.: Logical analysis of systems comprising feedback loops. *J. Theor. Biol.* 73(4), 631–656 (1978)
11. Govan, J., Deretic, V.: Microbial pathogenesis in cystic fibrosis: mucoid *pseudomonas aeruginosa* and *burkholderia cepacia*. *Microbiol. rev.* 60(3), 539–574 (1996)
12. Guespin-Michel, J., Kaufman, M.: Positive feedback circuits and adaptive regulations in bacteria. *Acta Biotheor.* 49(4), 207–218 (2001)

Modeling of Genetic Regulatory Network in Stochastic π -Calculus

Mylène Maurin, Morgan Magnin, and Olivier Roux

IRCCyN, CNRS UMR 6597,
1 rue de la Noë, BP 92101, F-44321 Nantes Cedex 3, France
{mylene.maurin,morgan.magnin,olivier.roux}@irccyn.ec-nantes.fr

Abstract. In this paper, we address the problem of modeling biological regulatory networks thanks to the stochastic π -calculus. We propose a method which extends a logical method, that is the approach of René Thomas. By introducing temporal and stochastic aspects there, we make our formalism closer to biological reality. We then use the SPiM stochastic simulator to illustrate the practical interests of this description. The application example concerns the behaviors of four interacting genes involved in the λ -phage. Interesting results are emerging from the simulations. First, it confirms knowledge of the regulation phenomena. In addition, experiments with different values of the delay parameters give some precious hints of a tendency either for the lytic phase or to the lysogenic phase.

Keywords: Genetic regulatory network, Stochastic π -calculus, System biology.

1 Introduction

1.1 Modeling Biological Systems

Interactions between genes and products of genes (*i.e.* proteins) are central to many cellular processes (e.g. cell differentiation or temperature control) in eukaryotes and prokaryotes cells. These processes can be modeled thanks to genetic regulatory networks, that represent a set of genes and proteins with their respective interactions. The correct modeling of biological networks is fundamental to acquire a better understanding of the living; *in silico* experiments can help to predict, achieve and understand *in vitro* and *in vivo* experiments.

System biology is a still recent research field. It focuses on the study of the interactions between the components of biological systems and how these interactions guide the behavior of the system. Due to their complexity, regulatory networks are often difficult to understand intuitively. This why theoretical models and computational tools are required. Numerous formalisms were already defined. Hidde de Jong proposed a comprehensive description of these formalisms in [1].

The regulation process generally involves many proteins that concurrently control the evolution of the system (e.g. according to their concentration). Such

a framework implies we take into account the two following key factors: stochasticity and evolution rate of each actor.

First, the value of several parameters can widely vary from a cell to another [2]. In addition, if an organism follows one specific development cycle, an other organism with slightly different physiological values may follow an other cycle. It is thus essential to have a snapshot of the global variety of interactions between cells, proteins and genes in order to perform a correct analysis of a biological system. Then a probabilistic approach is of much interest.

An other major topic is the integration of timing information in the models. The theory of chemical kinetics defines that a protein involved in regulation phenomena is active once it has reached a threshold. When many proteins are involved, we want to take into account their respective evolution rate to determine which one reaches its level first and then modifies all the behavior of the system.

1.2 Related Work

Several formalism have been proposed for the modeling of genetic regulatory networks as boolean networks, Petri nets, hybrid automata, hcc or reaction rules. We briefly describe now some of them.

Petri Nets and Hybrid Petri Nets. Biological systems are typical systems where the notion of concurrency is central. When modeling a gene regulatory network, one has to represent the respective evolution of the different actors of the system. Petri nets are one of the efficient formalisms that allow to concisely model concurrency [3]. Various extensions of Petri nets have been proposed to model and verify hybrid or timed systems. In the field of systems biology, Petri nets have been preferably used to simulate systems [4] compared to hybrid Petri nets. Nevertheless, as this formalism offer extensions like stochastic Petri nets to encompass stochastic behaviors, it appears very promising to lead more comprehensive studies on biological networks. Works have been recently published to show how Petri nets may be used to integrate both qualitative and quantitative analysis: in [5], the authors propose a Petri net based framework to draw a link between qualitative, stochastic and continuous paradigms. They yet recognize that further research is needed to precisely understand the relationships between the respective properties of each description.

Timed and Hybrid Automata. Timed automata [6] and hybrid automata [7,8] have been used in various approaches to refine pure boolean or discrete modeling frameworks such as that of René Thomas. Mainly, the basic ideas of those timed or hybrid modelings lie in the principle that the evolutions of genes expression levels may no longer be considered as instantaneous. Since these changes take some time, they have to be taken into account and it has some relevance. Timed systems reveal to be well adapted to represent the dynamics in these phases. It results in more accurate behavior predictions.

Pathway Logic. This is an approach based on rewriting logic [9,10]. In [9], the authors develop qualitative models of metabolic and signaling processes. In this

formalism, a metabolic network is transformed into a system of symbolic differential equations using variables with an a priori unknown rate. This approach aims at studying how a system could evolve using techniques based on logical inference.

Reaction Rules. With this formalism [11], a model is defined by a set of reaction rules. Kinetic expressions, parameters values or initial conditions can be associated to the reactions. In order to model biochemical systems, a language based on temporal logic for the biological properties, and on Systems Biology Markup Language (SMBL) for modeling the elementary interactions between molecular species, has been developed: BIOCHAM [12].

BioAmbients. BioAmbients [13] has been developed to model biological compartments. This corresponds to an extension of the mathematical domain of the stochastic π -calculus with additional entities. BioAmbients calculus is suitable for modeling the movement of molecules, the dynamic rearrangement of cellular compartments, and the interaction between molecule inside a compartment.

Our aim differs from this work as we are mostly involved in regulation modeling rather than in molecules movements.

The Stochastic π -calculus. In this work, we propose to model genetic regulatory network in the stochastic π -calculus. This is a process algebra which allows us to assign rates to actions (communications or silent moves), and is effective to model and simulate a wide range of biological systems. Although the π -calculus was originally developed for specifying concurrent computational systems, Aviv Regev and al. suggest to use it to model biological systems - and particularly the RTK-MAP Kinases cycle [14]. Since these first uses in a biological context, many regulatory networks have been modeled with the stochastic π -calculus, e.g. gene regulation by positive feedback [15], cell cycle control in eukaryotes [16] or λ -phage gene regulation with two genes [17].

1.3 Our Contribution

The aim of this paper is to present a formalism based on the stochastic π -calculus that is both enough realistic to describe biological systems and enough simple to be generalized to other regulatory networks.

In this paper, we focus on the λ -phage example [18,19,20,21] in its most relevant model, that is with four genes. The formalism we introduce can be seen as an extension of the qualitative approach of René Thomas. After having theoretically defined our model, we simulate practically it thanks to the Stochastic Pi-Machine (SPiM) developed by Phillips and Cardelli [22]. As numerous quantitative and qualitative data are available for the λ -phage, we are able to check and discuss the correctness of our approach. We obtain very promising results that not only confirm what we already knew about the regulation process but also lead to knowledge about the tendency either for the lytic phase or the lysogenic phase.

The originality of our work lies in the introduction of temporal and stochastic aspects to improve a purely discrete formalism and to be thus closer to biological reality.

1.4 Outline of the Paper

The paper is organized as follows: section 2 gives an overview of the stochastic π -calculus and the Stochastic Pi-Machine (SPiM). In section 3, we present the extension of the qualitative approach of René Thomas which is based on the stochastic π -calculus and finally, in section 4, we apply this method to the biological system we study, the λ -phage gene regulation.

2 The Stochastic π -Calculus

In order to integrate both stochastic and temporal information in our models, we use the stochastic π -calculus as a formal language. The stochastic π -calculus is a process algebra created by Corrado Priami in 1995 [23]. It allows to describe the temporal evolution of *parallel systems* (this means systems composed of various autonomous components that are executed simultaneously and capable of exchanging information via communication channels). It extends the asynchronous π -calculus previously introduced by Milner, Parrow and Walker [24] by adding a notion of reaction rate. These rates are directly correlated with the probability of the occurrence of a reaction and the transitions have exponential distribution. As a consequence, the transition system is mapped into continuous time Markov chains.

The stochastic π -calculus allows one to formally model biological system. To simulate them, some tools have been developed : BioSpi [14], the Stochastic Pi-Machine (SPiM) [25] and the Stochastic Pi-Calculus with Concurrent Object (SPiCO) [17] are three of these tools. We choose to work with SPiM whose performances seemed very promising.

2.1 The Stochastic Pi-Machine: SPiM

To simulate our models, we use the Stochastic Pi-Machine (SPiM) developed by Andrew Phillips and Luca Cardelli since 2004 [25]. The underlying algorithm is based on the standard theory of chemical kinetics using an adaptation of the Gillespie algorithm [26]. At each step the algorithm chooses one of the possible reactions where the probability of the reaction is proportional to its rate. The specification of the machine has been proved correct with respect to the calculus [25].

SPiM Language

We now present the key elements of the SPiM language. Its syntax and its corresponding graphical representation are summarized in Figure 1 and Figure 2.

The *neutral element* is the null process. It can also be denoted 0. An *action* is either a stochastic delay or an emission or a reception of a message on a

Action $\pi ::=$		
$!x(m)$	$?x(n)$	τ_r
sends value m on channel x	Receives value n on channel x	Delay at rate r
$!x(m) \downarrow$	$?x(n) \downarrow$	$\tau_r \downarrow$

Fig. 1. Overview of the π -calculus syntax (1)


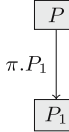
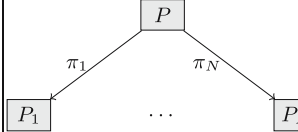
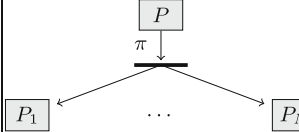
Process $P ::=$			
0	$\pi.P_1$	$\pi_1.P_1 + \dots + \pi_N.P_N$	$\pi.(P_1 \dots P_N)$
Nul process	makes an action	Chooses between actions	Parallel composition of processes
			

Fig. 2. Overview of the π -calculus syntax (2)

channel. *Operators* can be parallel composition, non-deterministic choices, replication of a process. By lack of space, we omit the exhaustive formal definition of the stochastic π -calculus, but we give a summary of the calculus syntax. The complete language is defined in [27].

The stochastic π -calculus thus allows to express the behaviors of a biological system thanks to simple computational components. These ones are autonomous; the only factors that influence their evolution are emission and reception via communication channels.

3 Our Formalism: An Extension of the Model of René Thomas

This modeling is based on the multi-valued approach of René Thomas extended by Denis Thieffry [28] in which biological regulatory networks are represented by a graph and a set of logical parameters.

1. A graph G is usually defined by two components:
 - a set of vertices V representing the genes of the network.
 - a set of oriented edges E representing the interactions where each edge is labeled by a couple (s, α) where s is an integer, called qualitative threshold and $\alpha \in \{+, -\}$ is the sign of the regulation.

We put an arc between a vertex v_1 and a vertex v_2 if the protein synthesized from gene v_1 exercises a control on the gene v_2 . We call v_1 a *resource* of v_2 if v_1 induces an increase of v_2 , i.e if the regulation $v_1 \rightarrow v_2$ is positive (resp. negative), the regulation must be active (resp. inactive).

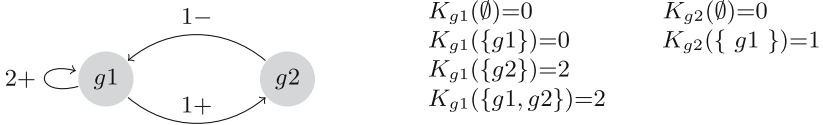


Fig. 3. An example with two genes

We define a state q of the system as a vector constituted by the qualitative levels of genes.

2. The logical parameter $K_v(\{\omega_v(q)\})$ is defined by the threshold towards which the gene v tends to evolve when its resources are $\omega_v(q)$ and the system is in the state q .

Example

Let us illustrate these notions with an example. The interaction graph and the logical parameters are given in Figure 3.

In this section, we propose a model in the stochastic π -calculus which translates the informations of an interaction graph and a set of logical parameters by adding an evolution rate to the initial model of René Thomas.

Given a genetic regulatory network with n genes modeled according to the formalism of René Thomas, we build a model in the stochastic π -calculus which encompasses $2n$ processes: n processes describe the temporal evolution of each gene and the n others translate the logical parameters.

In the case of our example, we define four processes. Let us detail the stages of the different processes.

The qualitative level of g_1 belongs to the discrete interval $\{0,1,2\}$. The process describing g_1 will be composed of three stages: g_1 level 0, g_1 level 1 and g_1 level 2.

The subset of logical parameters related to g_1 is composed of four $K_{g1}(\{\omega_{g1}(q)\})$. Thus, the process describing this subset will be built with four stages.

Similarly, the process describing g_2 will be composed of two stages and the process describing its set of logical parameters by two stages according to the table 3.

Let us now focus on the way they communicate. We define the following channels:

- $r_{g1}^{1+}, r_{g1}^{2+}, r_{g1}^{0-}$ and r_{g1}^{1-} are channels used to move the level of g_1 . In other words, r_{g1}^{0-} means "the level of gene g_1 decreases towards level 0".
- r_{g2}^{1+} and r_{g2}^{0-} are channels used to move the level of g_2 .
- $res_{g1}^{g1}, res_{g1}^{g2}, nores_{g1}^{g1}$ and $nores_{g1}^{g2}$ are channels used to indicate to which g_1 is resource or not. In other words, res_{g1}^{g2} means " g_1 becomes (or stays) resource to g_2 ".
- $res_{g2}^{g1}, res_{g2}^{g2}, nores_{g2}^{g1}$ and $nores_{g2}^{g2}$ are channels used to indicate to which g_2 is resource or not.

We construct our model according to the following steps:

1. Take into account the information of the logical parameters.

At each stage of the process describing the subset of logical parameters, a message is sent on a channel r_y^x .

Example: We have $K_{g_2}(\{g_1\})=0$, thus at the stage $K_{g_2}(\{g_1\})$ of the process describing the subset $K_{g_2}(\{\omega_{g_2}(q)\})$, a message on channel $r_{g_2}^{0-}$ will be sent.

2. Gene evolution.

The gene g_1 (resp. g_2) will receive the message delivered on channel $r_{g_1}^x$ (resp. $r_{g_2}^{x'}$) and then will evolve towards level x (resp. x'). While evolving, it sends messages on channels $(no)res_{g_1}^{g_1}$ and $(no)res_{g_1}^{g_2}$ (resp. $(no)res_{g_2}^{g_1}$ and $(no)res_{g_2}^{g_2}$).

Example: When g_1 increases from level 0 to level 1, the channel that are used are $nores_{g_1}^{g_1}$ and $res_{g_1}^{g_2}$.

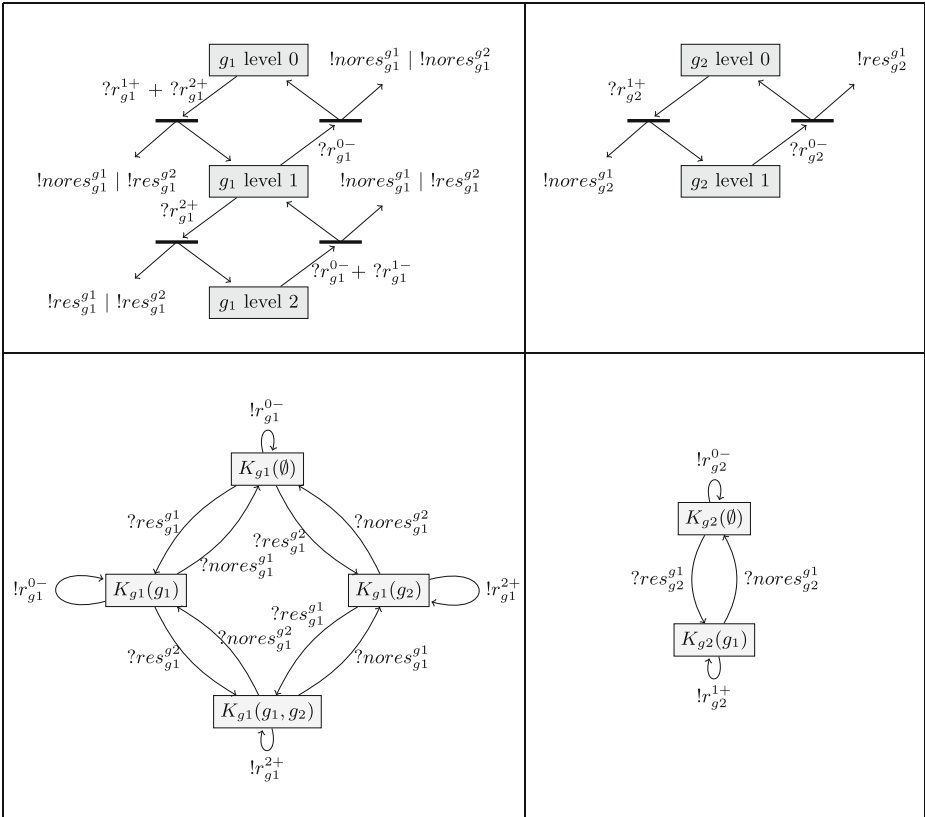


Fig. 4. The four processes that are necessary to model the example with two genes

3. Logical parameters evolution.

The resources messages will be received by the logical parameters which will evolve.

Example: If the process $K_{g_2}(\{\omega_{g_2}(q)\})$ is in state $K_{g_1}(\{g_1\})$ and a message is sent on channel $res_{g_1}^{g_2}$, the process will evolve towards $K_{g_1}(\{g_1, g_2\})$. Now, we can return to the first step.

Figure 4 shows the four processes concerning the example with two genes.

Stochastic and Temporal Aspects in our Formalism

To simulate the models in stochastic π -calculus, the algorithm of Gillespie is used. This one indicates that the probability that a reaction (i.e. the crossing of a threshold) occurs is directly correlated, according to an exponential distribution, with the time taken by this reaction to occur.

We introduce stochastic and temporal aspects in the formalism of René Thomas by means of the rates of channels. Indeed, rates linked to channels r_y^x allows us to mime the fact that a gene does not reach a threshold instantaneously but with a certain delay.

A contrario, we do not act on the rates of channels $(no)res_y^{y'}$ because when a gene reaches a threshold, the state of the system changes and so the processes describing the logical parameters must immediately change. We put an infinite rate to these channels, this implies that the probability that a reaction occurs (when a message is sent) is equal to 1.

4 Application to the λ -Phage

The λ -phage is a double-stranded DNA virus of the bacteria *Escherichia Coli* which has two possible evolutions:

- Lysogenic cycle: viral DNA may integrate itself into the bacteria DNA.
- Lytic cycle: the virus multiplies itself and lyses the bacteria which dies. The viruses are released and may infect others bacteria.

During the last decades, many works focused on the mechanisms of genetic regulatory of the λ -phage [18,21,19,20,21]. Many quantitative and qualitative data are available on the switch of the λ -phage. Since we intend to take time delays into account, this regulation process appeared particularly interesting.

The model of René Thomas is available in [28]. Let us focus on the biological regulatory graph (see Figure 5) and the set of logical parameters (see table 1)

Stochastic Parametrization

We introduce time and stochasticity as described above.

In order to evaluate the role of the rates on the final behavior, we consider only two rates: the first is linked to channels r_y^{x+} while the second is linked to channels r_y^{x-} . We have tested different rates for these channels and lead multiple simulations.

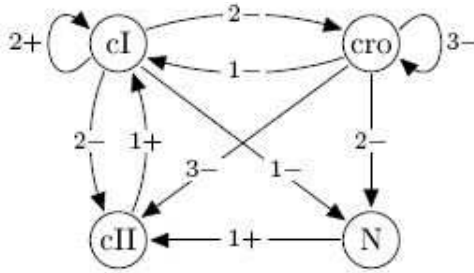


Fig. 5. Biological regulatory graph

Table 1. Logical parameters

$K_{cI}(\emptyset)=0$	$K_{cro}(\emptyset)=0$	$K_{cII}(\emptyset)=0$
$K_{cI}(\{cI\})=0$	$K_{cro}(\{cI\})=2$	$K_{cII}(\{cI\})=0$
$K_{cI}(\{cro\})=2$	$K_{cro}(\{cro\})=0$	$K_{cII}(\{cro\})=0$
$K_{cI}(\{cII\})=2$	$K_{cro}(\{cI,cro\})=3$	$K_{cII}(\{N\})=0$
$K_{cI}(\{cI,cro\})=2$	$K_N(\emptyset)=0$	$K_{cII}(\{cI,cro\})=0$
$K_{cI}(\{cI,cII\})=2$	$K_N(\{cI\})=0$	$K_{cII}(\{cI,N\})=0$
$K_{cI}(\{cro,cII\})=2$	$K_N(\{cro\})=0$	$K_{cII}(\{cro,N\})=0$
$K_{cI}(\{cI,cro,cII\})=2$	$K_N(\{cI,cro\})=1$	$K_{cII}(\{cI,cro,N\})=1$

The results presented in Figures 6 and 7 were obtained with $r_y^{x+} = 0.01s^{-1}$ and $r_y^{x-} = 0.1s^{-1}$. We start the simulation with all the genes at level 0 and we follow the evolution of processes describing the genes.

4.1 Results

In Figure 6, we show a simulation leads to the stable state corresponding to lysogenic pathway. In Figure 7, we show a simulation leading to the cycle corresponding to lytic pathway. In these figures, study focus on the threshold reached by the genes during the time of the simulation. When we simulate this model, the system always goes in one of the two stable states described in 28 by René Thomas and Denis Thieffry. By testing different values for the ratio between the rate of r_y^{x+} and the rate of r_y^{x-} , we noticed that the system goes more often in one pathway or in the other one according to the value of this ratio. The first observation leads us to introduce the following property: when the ratio between these rates is high (resp. low), the system tends to enter into the lysogenic state (resp. lytic cycle).

4.2 Toward a Probabilistic Reasoning

We are currently processing a statistical analysis to determine the exact role of this ratio for the choice between lytic cycle and lysogenic state. In Figure 8, we can see the proportion of simulations that leads to lysogenic stable state according to the ratio of rates. Each point corresponds to 300 simulations reduced

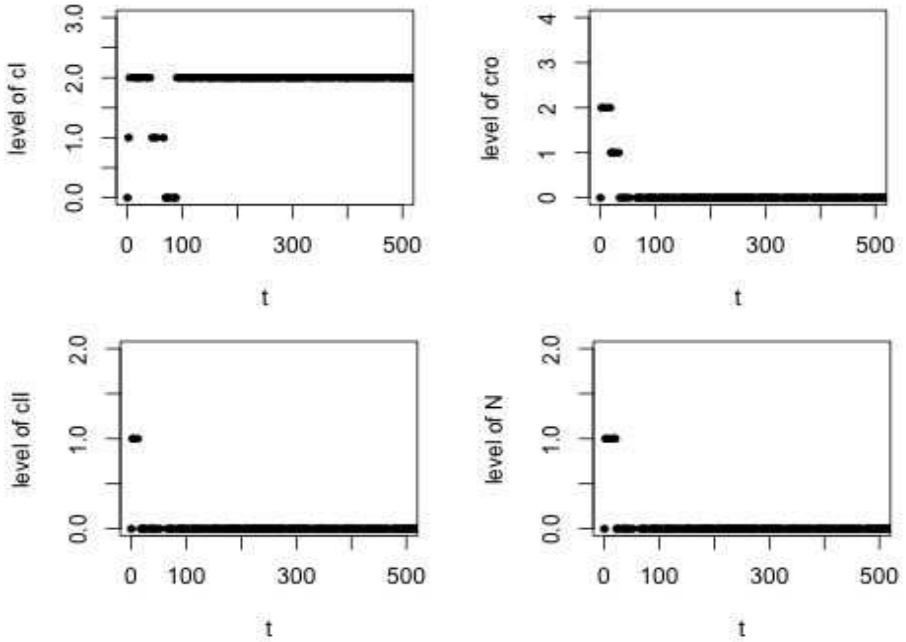


Fig. 6. Simulation results corresponding to lysogenic state

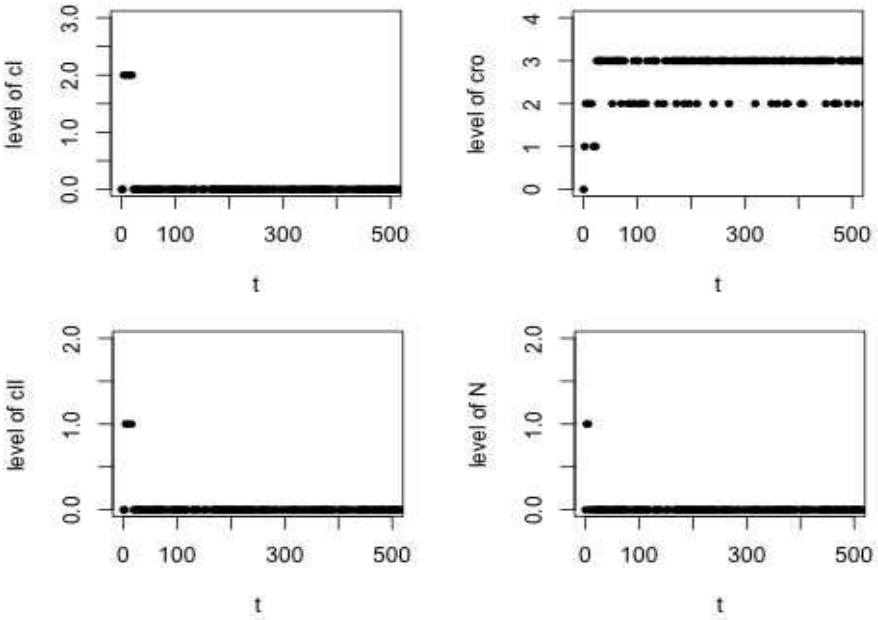


Fig. 7. Simulation results corresponding to lytic cycle

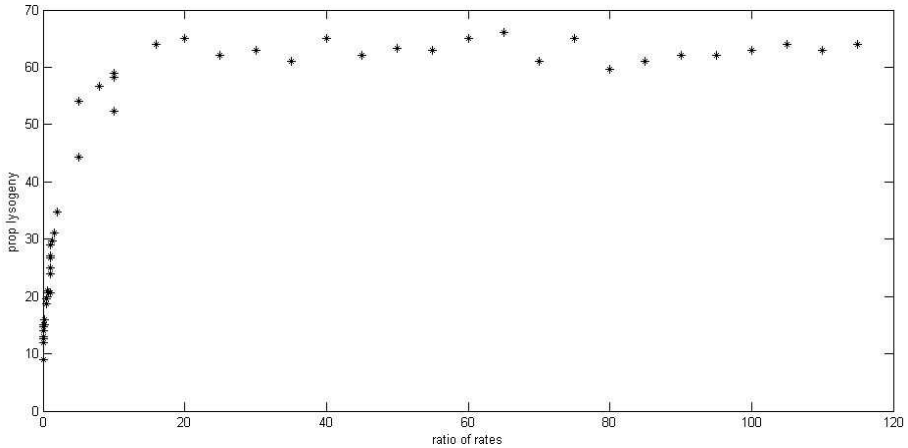


Fig. 8. Proportion of simulations that leads to lysogenic stable state according to the ratio of rates

to 100. This figure shows that the ratio of rates plays a real role in the probability to reach a final stable state rather than the other. We also point out that it is not necessary to increase the ratio above the value 30. Indeed, the proportion of lysogenic state becomes stable for a ratio higher than 30. The capacity of inferring properties on the system thanks to the model is a major advantage: it reinforces the relevance of our choice to introduce the temporality into the René Thomas modeling approach.

5 Conclusion

In this paper, we have considered the λ -phage gene regulation phenomena. We proposed a model in stochastic π -calculus which extends the approach of René Thomas by adding temporal and stochastic aspects.

We showed that our method leads to results that are consistent with the expected behavior of the regulation system. Moreover, we put the light on the fact that the time and the stochasticity give a major improvement to the discrete approach. We are currently working on a refinement of the model in order to determine precisely which rates of channels are significant in the choice of the final state.

Further work consists in developing a powerful tool capable to automatically model and simulate the behavior of biological regulatory networks with stochastic π -calculus. Thanks to such a tool, we will be able to perform statistical and probabilistic analyses. This will result in acquiring interesting hints for leading biological experimentations.

Acknowledgments. We are indebted to Damien Eveillard with whom we had enriching discussions.

References

1. De Jong, H.: Modeling and simulation of genetic regulatory systems: a literature review. *J. Comput. Biol.* 9(1), 67–103 (2002)
2. Arkin, A., Ross, J., McAdams, H.H.: Stochastic kinetic analysis of developmental pathway bifurcation in phage λ -infected *Escherichia coli* cells. *Genetics* 149, 1633–1648 (1998)
3. Petri, C.A.: Kommunikation mit Automaten. PhD thesis, Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2 (1962); 2nd edn., New York: Griffiss Air Force Base, Technical Report RADC-TR-65-377, vol. 1(suppl. 1), English translation (1966)
4. Chaouiya, C., Remy, E., Thieffry, D.: Petri net modelling of biological regulatory networks. *J. of Discrete Algorithms* 6(2), 165–177 (2008)
5. Heiner, M., Gilbert, D., Donaldson, R.: Petri nets for systems and synthetic biology. In: Bernardo, M., Degano, P., Zavattaro, G. (eds.) SFM 2008. LNCS, vol. 5016, pp. 215–264. Springer, Heidelberg (2008)
6. Siebert, H., Bockmayr, A.: Incorporating time delays into the logical analysis of gene regulatory networks. In: Priami, C. (ed.) CMSB 2006. LNCS (LNBI), vol. 4210, pp. 169–183. Springer, Heidelberg (2006)
7. Adélaïde, M., Sutre, G.: Parametric analysis and abstraction of genetic regulatory networks. In: Proc. 2nd Workshop on Concurrent Models in Molecular Biol. (Bio-CONCUR 2004), London, UK. Electronic Notes in Theor. Comp. Sci. Elsevier, Amsterdam (2004)
8. Ahmad, J., Bernot, G., Comet, J.P., Lime, D., Roux, O.: Hybrid modelling and dynamical analysis of gene regulatory networks with delays. *ComplexUs* 3(4), 231–251 (2007)
9. Eker, S., Laderoute, K., Lincoln, P., Talcott, C.: Pathway logic: executable models of biological networks. In: Fourth International Workshop on Rewriting Logic and Its Applications (WRLA 2002), Elsevier, Amsterdam (2002)
10. Talcott, C.: Formal executable models of cell signaling primitives. In: Margaria, T., Philippou, A., Steffen, B. (eds.) 2nd International Symposium On Leveraging Applications of Formal Methods, Verification and Validation ISOLA 2006, pp. 303–307 (2006)
11. Calzone, L., Fages, F., Soliman, S.: Biocham: an environment for modeling biological systems and formalizing experimental knowledge. *Bioinformatics* 22(14), 1805–1807 (2006)
12. Calzone, L., Chabrier-rivier, N., Fages, F., Soliman, S., Rocquencourt, I., Conrantes, P.: A machine learning approach to biochemical reaction rules discovery. In: Doyle III, F.J. (ed.) Proceedings of Foundations of Systems Biology and Engineering FOSBE 2005, pp. 375–379 (2005)
13. Regev, A., Panina, A., Silverman, W., Cardelli, L., Shapiro, E.: Bioambients: An abstraction for biological compartments. Elsevier Science, Amsterdam (2003)
14. Regev, A., Silverman, W., Shapiro, E.: Representation and simulation of biochemical processes using the pi-calculus process algebra. In: Proc. Pacific Symp. of Biocomputing, vol. 6, pp. 459–470 (2001)
15. Priami, C., Regev, A., Shapiro, E.Y., Silverman, W.: Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Inf. Process. Lett.* 80(1), 25–31 (2001)
16. Lecca, P., Priami, C.: Cell cycle control in eukaryotes: A biospi model. *Electr. Notes Theor. Comput. Sci.* 180(3), 51–63 (2007)

17. Kuttler, C., Niehren, J.: Gene regulation in the pi calculus: Simulating cooperativity at the lambda switch. *Transactions on Computational Systems Biology* VII 4230, 24–55 (2006)
18. Ptashne, M.: *A genetic switch: Phage λ and higher organisms*, 2nd edn. Cell Press and Blackwell science, Malden (1992)
19. Ackers, G.K., Johnson, A.D., Shea, M.A.: Quantitative model for gene regulation by λ phage repressor. *Proceedings of the National Academy of Science USA* 79, 1129–1133 (1982)
20. Shea, M.A., Ackers, G.K.: The or control system of bacteriophage lambda: a physical-chemical model for gene regulation. *J. Mol. Biol.* 181, 211–230 (1985)
21. Sauer, R.T.: *Molecular characterization of the lambda repressor and its gene ci*. Harvard University Press, Cambridge (1979)
22. Phillips, A., Cardelli, L.: *Spim* (2007), <http://research.microsoft.com/~aphillip/spim/>
23. Priami, C.: Stochastic π -calculus. *Computer Journal* 6, 578–589 (1995)
24. Milner, R.: A calculus of mobile processes. *Information and computation* 100, 1–77 (1992)
25. Phillips, A., Cardelli, L.: Efficient, correct simulation of biological processes in the stochastic pi-calculus. In: Bošnački, D., Edelkamp, S. (eds.) *SPIN 2007*. LNCS, vol. 4595, pp. 184–199. Springer, Heidelberg (2007)
26. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry* 81(25), 2340–2361 (1977)
27. Phillips, A.: *The SPIM Language*. Version 0.05 (2007)
28. Thieffry, D., Thomas, R.: Dynamical behaviour of biological regulatory networks - ii. immunity control in bacteriophage lambda. *Bull. Math. Biol.* 57(2), 277–297 (1995)

fMRI Activation Detection by MultiScale Hidden Markov Model

Fangyuan Nan¹, Yaonan Wang¹, and Xiaoping Ma²

¹ College of Electrical and Information Engineering, Hunan University,
Changsha, China

² School of Information and Electrical Engineering, China University of Mining and
Technology
nanfangy@hotmail.com

Abstract. This paper considers detection of functional magnetic resonance images (fMRIs), that is, to decide active and nonactive regions of human brain from fMRIs. A novel two-step approach is put forward that incorporates spatial correlation information and is amenable to analysis and optimization. First, a new multi-scale image segmentation algorithm is proposed to decompose the correlation image into several different regions, each of which is of homogeneous statistical behavior. Second, each region will be classified independently as active or inactive using existing pixel-wise test methods. The image segmentation consists of two procedures: edge detection followed by label estimation. To deduce the presence or absence of an edge from continuous data, two fundamental assumptions of our algorithm are 1) each wavelet coefficient is described by a 2-state Gaussian Mixture Model (GMM); 2) across scales, each state is *caused* by its parent state, hence the name of multiscale hidden Markov model (MHMM). The states of Markov chain are unknown ("hidden") and represent the presence (state 1) or absence (state 0) of edges. Using this interpretation, the edge detection problem boils down to the posterior state estimation given observation.

Keywords: functional magnetic resonance imaging (fMRI), wavelet analysis, image segmentation, edge detection, hidden Markov model, spatial-temporal modeling.

1 Introduction

1.1 Spatial Modeling of fMRI and Outline for the Method

Magnetic resonance imaging (MRI) is a powerful diagnostic imaging technique based on the principle of nuclear magnetic resonance, describing the interaction of nuclei and magnetic fields. While traditional MRI provides only static images to analyze anatomical structure, functional MRI (fMRI), a newer imaging modality which is based on MRI and just came to the stage around two decades ago, acquires a series of images to detect neural activity, to locate brain activation. In other words, the central task for fMRI is to obtain maps of active and nonactive regions of the brain [[1](#), [2](#), [3](#)].

Closely related to this paper is [1], which uses generalized likelihood ratio test to get the activation map.

As in [1], pixelwise detection for fMRI is most common in practice [3, 4]. However, these techniques do not take advantage of mutual information between neighboring pixels. Ignoring such spatial information can cause some problems. For example, at first sight to a physician, the last figure in [1] seems surprising because there are activation areas outside brain! On the other hand, utilizing spatial information may enhance our detection accuracy. For example, it may be quite possible that an activated (contiguous) area is larger than individual pixel dimensions. In other words, activated areas tend to occur in clusters of neighboring pixels; or if we know, by some means, there is strong indication that a large group of pixels, which may be thought of as one large pixel at very coarse (spatial) scale, is active, then the individual pixels inside this group, which may be regarded as pixels at a finer scale, are more likely to be active themselves. Hence comes the idea of (spatial) scale and incorporating spatial correlation into the fMRI detection process.

In light of the above simple idea, the kind of pixelwise detection is oversimplistic. Therefore it is necessary to develop detection methods taking advantage of spatial correlation. There are many approaches to attack the problem, for example, cluster analysis [6, 7, 5] and independent component analysis (ICA) algorithm [8]. Detection methods using Bayesian strategies have been proposed for fMRI [9, 10, 11]. Just as in pixelwise detection, we need to model each time series; when we turn attention to spatial correlation, we also need consider spatial modeling for our problem. This is by no means an easy job. [10, 11] use a Markov Random Field (MRF) model to determine the activation map.

But, we note that those Bayesian methods mentioned above are all restricted to modeling on the finest scale. Such methods tend to be very computationally demanding, and are often difficult to analyze and interpret. Therefore, multi-scale modeling (specifically, multiscale image segmentation) will be considered that incorporates spatial correlation information and is much more amenable to analysis and optimization.

Some work has already been done in this aspect [13, 14]. This paper will adopt a two-step approach for fMRI detection: multi-scale image segmentation will be first used to break the correlation image into several different regions, each of which is of homogeneous statistical behavior, then these regions will be classified independently as active or inactive by single-pixel detection methods. Since pixelwise detection has been elaborated in other literature, this paper will concentrate on the first step of image segmentation.

2 Image Segmentation by Multi-scale Hidden Markov Modeling of Wavelet Coefficients

The main ideas originate from [20]. This method consists of two procedures: the first one is edge detection and the other is label (state) estimation. The idea of applying wavelet analysis to edge detection is quite simple. Roughly speaking,

wavelet coefficients represent the differences between function approximations at different scales (or resolutions), one kind of differentiation, intuitively, is well suited for edge detection. In the following four subsections, I'll address in detail how to achieve the first step: edge-detection. Then I'll explain briefly the second step and how to apply our formulations in 1-D case to 2-D image.

2.1 Likelihood Function for the fMRI Data

Modelling the spatial correlation of the fMRI data in making statistical inference is a challenging problem for which not many solutions have been proposed than the solutions addressing the temporal dependencies. In this paper, we deal with fMRI magnitude images. As noted in the begining, the fMRI correlation image in this case may be modeled as a 2-d Gaussian process. The correlation between reference, which is characteristic of the BOLD response [11] and assumed known in this paper, and the magnitude time series $\mathbf{z}: c = \mathbf{r}^T \mathbf{z} = \sum_{j=1}^N r_j z_j$ is Gaussian distributed (provided the signal noise ratio is not very small). But for simplicity, let us first consider the one-dimensional analog. Yielding to convention, we assume that the length of the correlation sequence is a power of 2. The observation model is:

$$c_k^J = \rho_k^J + w_k^J, \quad k = 0, \dots, 2^J - 1, \tag{1}$$

where $\mathbf{c}^J = \{c_k^J\}$ are the observations (spatial correlations), $\rho^J = \{\rho_k^J\}$ are "true" correlation values, and $\{w_k^J\}$ are noise.

Now we are going to use a special (the simplest) multi-scale analysis, i.e., Haar wavelet transform on the data:

$$c_k^j = \frac{c_{2k}^{j+1} + c_{2k+1}^{j+1}}{\sqrt{2}}, \quad k = 0, \dots, 2^j - 1, \quad J_0 \leq j \leq J - 1.$$

The multi-scale analysis of the data ρ and \mathbf{w} is defined in an analogous way.

It's then straightforward to see that

$$c_k^j = \rho_k^j + w_k^j.$$

The noise $\{w_k^j, k = 0, \dots, 2^j - 1\}$ are assumed to be (spatially) independent, identically distributed zero-mean Gaussian random variables with variance σ^2 . It then follows that the preceding sentence is also true for any j , resulting in the likelihood function

$$p(\mathbf{c}^j | \rho^j) = \prod_{k=0}^{2^j-1} \mathcal{N}(c_k^j | \rho_k^j, \sigma^2), \quad J_0 \leq j \leq J \tag{2}$$

where $\mathbf{c}^j \equiv \{c_k^j\}_{k=0}^{2^j-1}$ and similarly for ρ^j , $\mathcal{N}(x | \rho, \sigma^2)$ denotes a Gaussian density with mean ρ and variance σ^2 evaluated at the point x .

The relationship between a “parent” (*e.g.*, c_k^j) and a “child” (*e.g.*, c_{2k}^{j+1}) is very important in multi-scale data analysis. The parent-child conditional likelihood in our case turns out to be:

$$p(c_{2k}^{j+1} | c_k^j, \rho) = \mathcal{N} \left(c_{2k}^{j+1} | \frac{c_k^j}{\sqrt{2}} + \frac{\theta_k^j}{\sqrt{2}}, \frac{\sigma^2}{2} \right), \tag{3}$$

where the canonical parameter

$$\theta_k^j = \frac{\rho_{2k}^{j+1} - \rho_{2k+1}^{j+1}}{\sqrt{2}}, \tag{4}$$

is simply the Haar wavelet coefficient of true correlation ρ at scale j and location k . This nice form of the likelihood suggests the use of so called *conjugate prior* for the wavelet coefficients in the following subsection, which complements the observation model and leads to closed form for the posterior of the states.

Further, the likelihood function (2) with $j = J$ can be factorized as follows [2]:

$$p(\mathbf{c} | \rho) = p(\mathbf{c}^{J_0} | \rho^{J_0}) \prod_{j=J_0}^{J-1} \prod_{k=0}^{2^j-1} p(c_{2k}^{j+1} | c_k^j, \theta_k^j), \tag{5}$$

where J_0 is the coarsest scale for the analysis (usually we use $J_0 = 0$), $p(c_{2k}^{j+1} | c_k^j, \theta_k^j)$ is given by (3) and $p(\mathbf{c}^{J_0} | \rho^{J_0})$ is given by (2) with $j = J_0$.

2.2 Multi-scale Hidden Markov Model (MHMM) for the Prior of the Wavelet Coefficients

Now let’s consider prior (joint) probability for the (unknown) wavelet coefficients θ . A simple approach is to model them as independent Gaussian mixture random variables. We move beyond this simple prior, by specifying probabilistic dependencies between the *states* underlying the mixtures of parent and child wavelet coefficients. To deduce discrete state estimations from continuous data, the *key point* for our algorithm is to associate the *continuous* wavelet coefficients with a 2-state *discrete* Markov chain.

Specifically, for our real problem, the states (edge or smoothness) of the Markov chain are unknown (“hidden”) and represent the presence or absence of edges: state 0 indicates a homogeneous region, state 1 represents the existence of an edge. We perceive that the underlying signal is generally smooth with a few large edges, then the following modeling is intuitively reasonable, *i.e.*, we consider two-state mixture model where state ‘0’ is a highly probable low-variance Gaussian density, indicative of a homogeneous region, while state ‘1’, corresponding to another less likely Gaussian density with a larger variance, indicates the presence of an edge (non-smooth area). Using this interpretation, we may test for the presence of an edge simply by checking whether the following condition holds or not:

$$p(s_k^j = 1 | \mathbf{c}) > p(s_k^j = 0 | \mathbf{c}), \tag{6}$$

If it holds, we conclude there is an edge at scale j and location k . Otherwise, there is not.

Mathematically, the MHMM is based on the assumption that the value of each state s_k^j is *caused* by the value of its parent state. This leads to the factorization of the joint state probability function:

$$p(\mathbf{s}) = \prod_{j=J_0}^{J-1} \prod_{k=0}^{2^j-1} p(s_k^j | s_{\lfloor k/2 \rfloor}^{j-1}), \tag{7}$$

where $p(s_0^0 | s_0^{-1}) \equiv p(s_0^0)$. At the coarsest scale $j = 0$, we have no parent wavelet coefficient and so we introduce a prior for the state of the wavelet coefficient $p(s_0^0)$. Note that ρ_0^0 is the global average correlation data.

Another property of the HMM is that, given their respective state values, all parameters θ are conditionally independent [25, 21]. That is,

$$p(\theta | \mathbf{s}) = \prod_{j=J_0}^{J-1} \prod_{k=0}^{2^j-1} p(\theta_k^j | s_k^j). \tag{8}$$

where the prior probability $p(\theta_k^j | s_k^j)$ is assumed to be Gaussian.

$$p(\theta_k^j | s_k^j = m) = \mathcal{N}(\theta_k^j | \mu_m^j, \tau_m^{j2}). \tag{9}$$

We regard the signal and its wavelet coefficients as realizations from a large family of random signal. Therefore, *collectively*, we assume $\mu_m^j = 0$.

2.3 Solution for Joint a Posterior Sate Probability

Having set up the formulations for likelihood and a *prior*, we are now ready to determine the *a. posterior* density of the *joint* states given observation. Note that:

$$\begin{aligned} p(\mathbf{s} | \mathbf{c}) &= \int p(\mathbf{s}, \theta | \mathbf{c}) d\theta \\ &\propto \int p(\mathbf{c} | \mathbf{s}, \theta) p(\theta | \mathbf{s}) p(\mathbf{s}) d\theta \\ &= \prod_{j=J_0}^{J-1} \prod_{k=0}^{2^j-1} \int p(c_{2k}^{j+1} | c_k^j, \theta_k^j, s_k^j) p(\theta_k^j | s_k^j) p(s_k^j | s_{\lfloor k/2 \rfloor}^{j-1}) d\theta_k^j \\ &= \prod_{j=J_0}^{J-1} \prod_{k=0}^{2^j-1} p(s_k^j | s_{\lfloor k/2 \rfloor}^{j-1}) L_k^j(s_k^j) \end{aligned} \tag{10}$$

where m_k^j is one particular state value assumed by random variable s_k^j and $L_k^j(s_k^j = m) \propto p(c_{2k}^{j+1} | c_k^j, s_k^j = m)$, the essential ingredients for our estimation

of the *a posteriori* states, are actually marginal likelihoods. From the likelihood function in (3) and the prior in (9), we derive them to be:

$$\begin{aligned}
 L_k^j(m) &= \int p(c_{2k}^{j+1} | c_k^j, \theta_k^j) p(\theta_k^j | s_k^j = m) d\theta_k^j \tag{11} \\
 &= \mathcal{N} \left(c_{2k}^{j+1} \middle| \frac{\mu_m^j}{\sqrt{2}} + \frac{c_k^j}{\sqrt{2}}, \frac{\tau_m^{j2} + \sigma^2}{2} \right), \\
 & \quad J_0 \leq j \leq J - 1, \quad k = 0, 1, \dots, 2^j - 1, \quad m = 0, 1. \tag{12}
 \end{aligned}$$

Proof, omitted. Refer to [2].

2.4 Marginal *a Posteriori* State Probability Calculation

With these formulations ready, we can use the upward-downward algorithm [20] to determine the most likely marginal *a posteriori* state for the wavelet coefficients θ_k^j and then use equation (6) to test the presence of an edge.

In the upward-downward algorithm, the **Up Step** marginalizes the joint posterior state probability recursively from the finest scale $j = J - 1$ to the coarsest scale $j = 0$. At the end the posterior state probabilities $\{p(s_0^0 = m | \mathbf{c})\}_{m=0}^{M-1}$ are provided and partial marginalizations are also stored for use in the Down Step. The **Down Step** computes the marginal posterior state probabilities for each s_k^j recursively. For the specific flow of the upward-downward algorithm, refer to [20].

2.5 Segmentation

After the edges are determined, it is straightforward to formulate likelihood ratio test to estimate the label (state) of each homogeneous region.

Consider the following multi-hypothesis problem. The observation $\mathbf{c} = [c_1 \ c_2 \ \dots \ c_n]^T$ within each homogeneous region is Gaussian random vector of dimension n . The M hypotheses are

$$H_i : \mathbf{c} \sim N(\mathbf{m}_i, \mathbf{C}_i), \quad i = 1, 2, \dots, M, \tag{13}$$

where \mathbf{m}_i and \mathbf{C}_i is the mean vector and covariance matrix of the observation under the i th $i = 1, 2, \dots, M$ hypothesis, which are assumed to be known. Suppose each hypothesis is equally likely and minimum error criterion is adopted [28], the decision rule then boils down to choosing H_j where

$$j = \arg \min_i \| \mathbf{c} - \mathbf{m}_i \|^2 + \ln |\mathbf{C}_i| \tag{14}$$

where $\| \mathbf{c} - \mathbf{m}_i \|^2 \equiv (\mathbf{c} - \mathbf{m}_i)^T \mathbf{C}_i^{-1} (\mathbf{c} - \mathbf{m}_i)$ and $|\mathbf{C}_i|$ is the determinant of \mathbf{C}_i .

In the following simulated processing, the observation within each homogeneous region is assumed to be independent and identically distributed (*i. i. d.*), that is, I assume $\mathbf{m}_i = m_i \mathbf{1}$ and $\mathbf{C}_i = \sigma_i^2$.

Some important practical problems are: how to assign the a prior probability for s_0^0 , how to assign transition probabilities for the states:

$$p\left(s_k^j = m \mid s_{\lfloor k/2 \rfloor}^{j-1} = m'\right). \tag{15}$$

and how to determine the parameters for the Gaussian distribution characterizing each homogeneous region. Theoretically, these parameters are estimated by a complicated E-M algorithm. For our initial investigation, we set them empirically (by observation). It turns out that our experiment results are insensitive to the *a priori* probability and transition probability. The robustness is a nice feature.

2.6 Extension to 2 Dimensions

We can extend the multi-scale analysis and MHMMs easily from 1-D sequence to 2-D images. Instead of taking the usual 2-D wavelet transform to the original image; we use the following conversion method. First we convert the original 2-D image into 1-D sequence, and then apply previous 1-D wavelet analysis to the resulting sequence. The conversion details are: first split the image vertically into two halves, then horizontally splitting each half into two quarters, and iterate until each one is a 1×1 pixel. The merit of this conversion is that it retains the original spatial configuration. Refer to figure (II) for details.

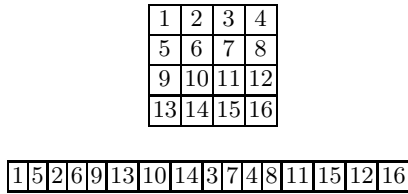


Fig. 1. Conversion of a 2-D image to 1-D sequence

2.7 One Simulation Testifying Image Segmentation by Our Algorithm

Figure (2) a)-b) shows a simulated noisy image, the grey level of the segmented image respectively. A two-state MHMM was specified for this problem with the following parameter settings:

$$\begin{aligned} \tau_0 &= 1, \\ \tau_1 &= 100, \\ \varrho_0^0(0) &= .9, \\ \varrho_k^j(0|0) &= .9, k = 0, \dots, 2^j - 1, j = 1, \dots, J - 1, \\ \varrho_k^j(0|1) &= .25, k = 0, \dots, 2^j - 1, j = 1, \dots, J - 1. \end{aligned}$$

Figure (b) demonstrates that the results are excellent.

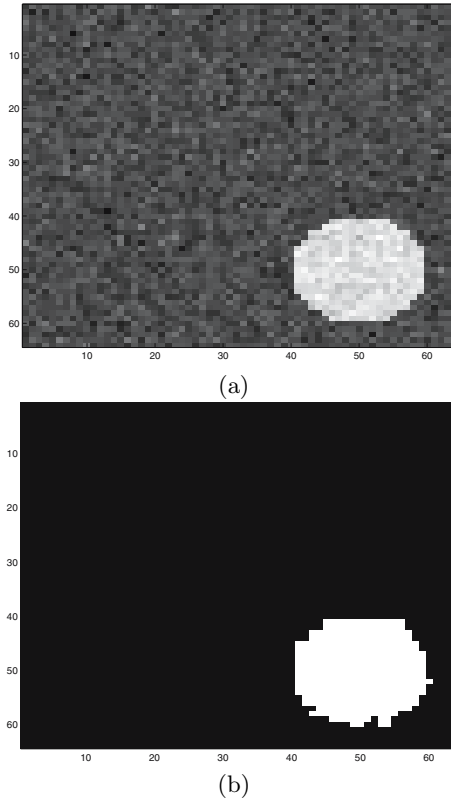


Fig. 2. a) Noisy image, b) Segmented Image

3 Processing Results for fMRI Simulated Data by Two-Step Approach

As stated in the beginning, the method for fMRI detection in this paper involves a two-step procedure: multiscale image segmentation will be first used to break the correlation image into different regions of homogeneous statistical behavior, each region will then be tested independently as active or inactive by single pixel detection method.

In order to see the potential of this method for fMRI detection, the following experiment is conducted to compare results from the combined effects of single pixel detection and image segmentation with results obtained based solely on pixel-wise detection.

Using the model in Equation (1) of [11], a simulated fMRI complex time series is generated at each pixel. In order to simulate the profile of the brain, the magnitudes of the baseline signal (a 's in Equation (1) in [11]) in the complex time series roughly follow the magnitude data from a static brain image. Actually the original complex data used in this example are exactly the same

as Figure (3b) in [11], which is reproduced here as Figure ((3a) for sake of comparison, Next, the correlation value at each pixel is computed by correlating the magnitude time series with the reference to produce Figure (3b).

Figure (3c) is the segmented result of the correlation image in Figure (3b) based on our algorithm. There are $M = 2$ labels: each pixel is assigned to either 0 or 1 according to its label. The parameters in this example are set to be:

$$\begin{aligned}\sigma^2 &= 1; \\ \tau_0^2 &= 1; \\ \tau_1^2 &= 100; \\ \varrho_0^0(0) &= .95; \\ \varrho(0|0) &= 0.95, k = 0, \dots, 2^j - 1, j = 1, \dots, J - 1; \\ \varrho(0|1) &= 0.05, k = 0, \dots, 2^j - 1, j = 1, \dots, J - 1; \\ m_0 &= 0; \\ m_1 &= 2.\end{aligned}$$

In this example, I set σ_1 and σ_0 (variances for the Gaussian distributions characterizing two homogeneous regions) to be equal. The test criterion in Equation (14) reduces to simpler form in this case. The original simulated active region in Figure (3a) is a 9*9 SQUARE; in Figure (3c) the white region is 10*8 RECTANGLE. They are in good agreement but not in perfect match. This is not surprising, since, in general, we cannot guarantee the segmentation step produces exactly the same GEOMETRY as the original simulated regions: what is shown here is just one realization of many random simulations.

Next I consider applying single pixel detection technique in [11] to each of the above homogeneously (statistically) distributed region. The idea is to regard each homogeneous region as one large, macro-pixel: the average of all time series inside each macro-pixel is taken to be the new time series characterizing this macro-pixel, then apply single pixel detection method (MC method in [11]) to the new time series individually to determine which of these macro-pixels is active and which one is inactive. By this approach, the micro-pixels (original pixels in Figure (3b)), in contrast to macro-pixel) corresponding to the black region in Figure (3c) are all inactive, which is expected since this region contains a large area outside the brain. The micro-pixels in Figure (3b) corresponding to the white region in Figure (3c) turns out to be all active. In other words, by this approach, only 9 pixels inside the square are missed while the 8 pixels outside the square are false-alarmed.

Now let us take a comparison between Figure (3a) and Figure (3c) in this paper. For the former, we see spurious activation regions outside the brain. However, the falsely alarmed regions disappear in Figure (3c) (except for 8 pixels outside the square) after combining image segmentation with single pixel detection. Pure single pixel detection methods failed to detect some active pixels inside the small square in last figure of [11]. However, these regions (except for 9 pixels) are now correctly detected by combining image segmentation with single pixel detection.

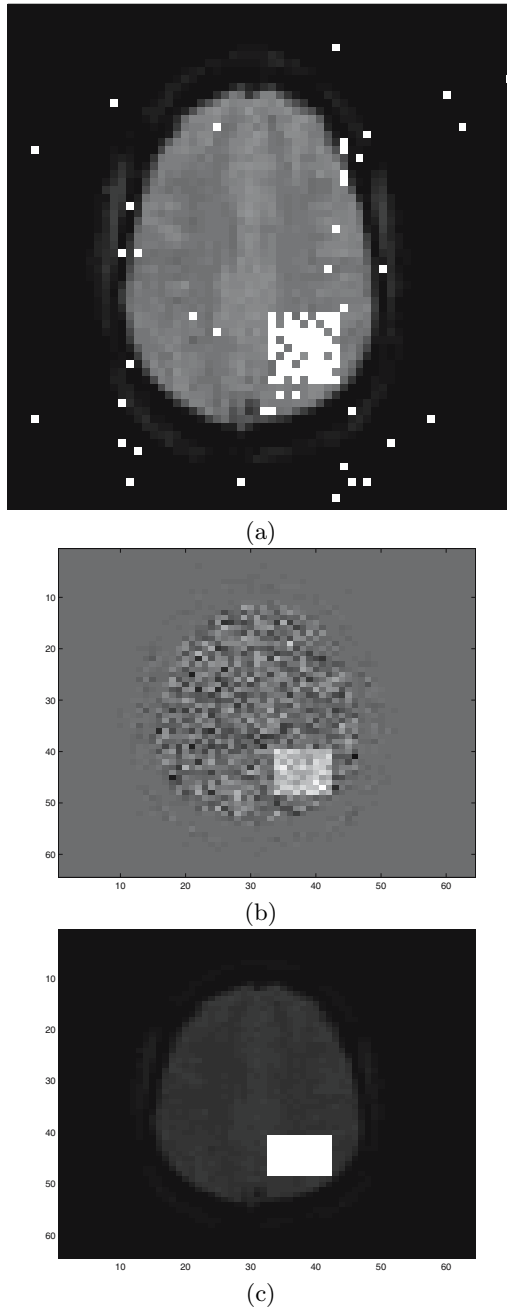


Fig. 3. Comparison of detection results from one step pixelwise method and from two-step approach. (a) Detection Image from [11] (b) fMRI correlation image, (c) Segmented image of (b), also final detection results by combination use of image segmentation and single pixel detection.

The enhancement of detection efficiency is clearly visible and also easily understandable. actually we are given spatial-temporal series. However, the pixel-wise detection method only takes temporal information into account: spatial information is completely ignored. The image segmentation algorithm in this paper exactly complements the pixel-wise detection and remedies its shortcoming: it utilizes the spatial correlation information inherent in the data. So it is no wonder that the detection performance improves after image segmentation.

One point to be noted is that in Figure (3c) the brain profiles are artificially overlapped, as are the cases in the last Figure in [1].

4 Conclusions and Discussions

fMRI signals are actually both temporally and spatially dependent. Pixel-wise detection, however, considers only temporal correlation information and ignores spatial correlation information. In order to remedy this deficiency, this paper uses a multi-scale image segmentation algorithm to first segment an fMRI correlation image into several regions, each with homogeneous statistical behavior. A single pixel detection algorithm is then applied to each homogeneous region. Extensive simulations demonstrate improved efficacy of our method.

Considering utilizing spatial correlation information, this paper uses Bayesian image segmentation method. Other approaches involving spatial consideration can be used as well. For example, clustering analysis are gaining more recognition in this field [5]. Formulation as decentralized detection problem is also a possible candidate [29].

References

1. Nan, F., Nowak, R.D.: Generalized likelihood Ratio Detection for fMRI using complex data. *IEEE Trans. Medical Imaging* 18(4), 320–329 (1999)
2. Nan, F.: Novel Detection Methods for functional magnetic resonance imaging, Ph. D. thesis, Dept. of ECE, Michigan State University (May 2000)
3. Bandettini, P.A., Jesmanowicz, A., Wong, E.C., Hyde, J.S.: Processing Strategies for time course data sets in functional MRI of the human brain. *Magnetic Resonance Medicine* 30, 161–173 (1993)
4. Genovese, C.R.: A time-course model for fMRI data. In: *Proc. Intl. Symp. Magn. Reson. Med. Meeting*, p. 1669 (1997)
5. Forman, S., Cohen, J.C., Fitzgerald, M., Eddy, W.F., Mintun, M.A., Noll, D.C.: Improved assesment of significant change in fucntional magnetic resoance fMRI: Use of cluster size threshold. *Magnetic Resonace Medicine* 33, 636–647 (1995)
6. Golay, X., Kollias, S., Stoll, G., Meier, D., Valavanis, A., Boesiger, P.: A new correlation -based fuzzy logic clustering algorithm for fMRI. *Magnetic Resonance, Medicine* 40, 249–260 (1998)
7. Goutte, C., Toft, P., Rostrup, E., Nielsen, F.A., Kai Hansen, L.: On clustering fMRI time series. *NeuroImage* 9, 298–310 (1999)
8. McKeown, M.J., Makeig, S., Brown, G.G., Jung, T.-P., Kindermann, S.S., Bell, A.J., Sejnowski, T.J.: Analysis of fMRI data by blind separation into independent spatial components. *Human Brain Mapping* 6, 160–188 (1998)

9. Frank, L.R., Buxton, R.B., Wong, E.C.: Probabilistic analysis and functional magnetic resonance imaging data. *Magn. Reson. Med.* 39, 132–148 (1998)
10. Kim, T., Al-Dayeh, L., Patel, P., Singh, M.: Bayesian processing for fMRI. In: *Proc. Intl. Soc. Magn. Reson. Med.* Sidney, Australia (1998)
11. Descombes, X., Kruggel, F., von Cramon, D.Y.: Spatial-temporal fMRI analysis using Markov random fields. *IEEE Trans. Medical Imaging* 17(6), 1028–1039 (1998)
12. Bouman, C., Shapiro, M.: A multiscale random field model for Bayesian image segmentation. *IEEE Trans. Image Processing* 3(2), 162–177 (1994)
13. Aston, J.A.D., Gunn, R.N., Hinz, R., Turkheimer, F.E.: Wavelet variance components in image space for spatiotemporal neuroimaging data. *NeuroImage* 25, 159–168 (2005)
14. Long, C., Brown, E.N., Manoach, D., Solo, V.: Spatiotemporal wavelet analysis for functional MRI. *NeuroImage* 23, 500–516 (2004)
15. Hossein-Zadeh, G., Soltanian-Zadeh, H., Ardekani, B.A.: Multiresolution fMRI activation detection using translation invariant wavelet transform and statistical analysis based on resampling. *IEEE Trans. Medical Imaging* 22(3) (March 2003)
16. Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and Bayesian restoration of images. *IEEE Trans. Patt. Anal. Mach. Intell.* 6(11), 721–741 (1984)
17. Perez, P., Heitz, F.: Restriction of Markov random field on a graph and multiresolution statistical image modeling. *IEEE Trans. Information Theory* 42(1), 180–190 (1996)
18. Malfait, M., Roose, D.: Wavelet-based image denoising using a Markov random field a priori model. *IEEE Trans. Image Processing* 6(4), 549–565 (1997)
19. Crouse, M.S., Nowak, R.D., Baraniuk, R.G.: wavelet based statistical signal processing using hidden Markov models. *IEEE Trans. Signal Processing* 46(4), 886–902 (1998)
20. Nowak, R.: Multiscale Hidden Markov Models for Bayesian Image Analysis. In: Vidakovic, B., Muller, P. (eds.) *Bayesian Interference in Wavelet Based Models*. Springer, New York (1999)
21. Rabiner, L.: A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 257–285 (1989)
22. Derin, H., Elliott, H.: Modeling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Trans. Patt. Anal. Mach. Intell.* 9(1), 39–55 (1987)
23. Luettggen, M.R., Karl, W.C., Willsky, A.S., Tenney, R.R.: Multiscale representations of Markov Random fields. *IEEE Trans. Signal Processing* 41(12), 3377–3396 (1993)
24. Forsgate, C.H., Krim, H., Irving, W.W., Karl, W.C., Willsky, A.S.: Multiscale segmentation and anomaly enhancement of SAR imagery. *IEEE Trans. Image Processing* 6(1), 7–20 (1997)
25. Deller, J., Proakis, J., Hanson, J.: *Discrete Time Processing of Speech Signals*. Prentice Hall, Englewood Cliffs (1993)
26. Rioul, O., Vetterli, M.: Wavelets and signal processing. *IEEE Signal Processing Magazine* 10, 14–38 (1991)
27. Vetterli, M., Kovacevic, J.: *Wavelets and Subband Coding*. Prentice Hall PTR, Englewood Cliffs (1995)
28. Srinath, M.D., Rajasekaran, P.K., Viswanathan, R.: *Introduction to Statistical Signal Processing with Applications*. Prentice Hall, Englewood Cliffs (1996)
29. Irving, W.W., Tsitsiklis, J.N.: Some properties of optimal thresholds in decentralized detection. *IEEE Trans. Automatic Control* 39(4), 835–838 (1994)
30. Mallat, S., Zhong, S.: Characterization of signals from multiscale edges. *IEEE Trans. Patt. Anal. Mach. Intell.* 14(7), 710–732 (1992)

cnF2freq: Efficient Determination of Genotype and Haplotype Probabilities in Outbred Populations Using Markov Models

Carl Nettelblad¹, Sverker Holmgren¹, Lucy Crooks², and Örjan Carlborg²

¹ Department of Information Technology, Uppsala University,
Box 337, SE-75105 Uppsala, Sweden
carl.nettelblad@it.uu.se

<http://www.it.uu.se/research/project/ctrail>

² Department of Animal Breeding and Genetics, SLU, Box 7023, SE-75007 Uppsala

Abstract. We have applied and implemented HMM (Hidden Markov Model) algorithms to calculate QTL genotype probabilities from marker and pedigree data in general population structures. These algorithms have a linear complexity in memory. In nearly all experimental pedigrees they result in more precise genotype estimates than the most commonly used approaches for determining genotypes at non-marker positions in QTL analysis in outbred F_2 line intercrosses [1], which include an exponential complexity factor as well as a data-reducing sampling step [2]. With a proper choice of parameters, the results from the existing methods can also be reproduced exactly. We show how the relative run times differ by a factor of 50 when 24 SNP markers are used, with our run time practically independent of marker count. The new method can also provide multi-generational probability estimates and perform haplotype inference from unphased data, which further improves accuracy and flexibility. An important future application of this method is for computationally efficient QTL genotype estimation in maps based on data from SNP chips containing 1000s of markers with mixed information content, for which there are no other suitable methods available at present.

1 Introduction

The development of dense single-nucleotide polymorphism (SNP) marker maps has resulted in a rapid increase in the number of genetic markers available for QTL (quantitative trait locus) analysis [3] using interval mapping (IM, [4]) techniques. IM schemes involve determining genotype probabilities at positions of incomplete marker information. The properties of new dense genetic maps imply that standard approaches for computing such probabilities, as e.g. described in [1] can now be drastically insufficient. The main objective of this paper is to present a class of algorithms that reduces the $O(m3^m)$ complexity for the established methods to $O(m \log m)$, where m is the number of genetic markers. We achieve this by realizing the underlying Hidden Markov Model (HMM) structure of the model used in the standard algorithm. In the software package R/qtI

[5], this type of approach has been used for data from specific cases of inbred populations, with similar computational complexity. We generalized upon this to include outbred data, as well as different population structures. We also show how the HMM approach can be further developed to enable haplotype inference. Producing phased, or haplotype, data can improve the accuracy of the genotype estimates and is also important for other forms of analysis. Furthermore, we also describe how to increase the accuracy for the computed genotype probabilities of multiple loci in the same linkage group by computing the specific joint probability. An implementation of the new algorithms is presented as the tool `cnF2freq` (same-Chromosome n -loci F_2 FREQUENCIES), as C++ source, with work undergoing to implement an R interface for multidimensional QTL searches. In the presentation below, we focus on data from F_2 populations, but as has already been remarked, our methods and implementations are designed to be applicable to general population structures.

1.1 Properties of Modern Marker Maps

The exponential complexity component of the algorithm described in [1] limits its practical use when there are many not fully informative markers (markers where the grandparent each allele was inherited from cannot be determined). Such is usual in analyses based on whole genome SNP (single-nucleotide polymorphism) chips. SNPs are attractive from a cost standpoint, and frequent in most genomes, at a rate of about one polymorphism per 1000 base pairs [6]. One drawback of SNPs is the limited variability in the individual markers. There are at most 4 possible values, corresponding to the nucleotides of the genetic code. However, most natural SNPs are biallelic, one wildtype allele and one mutant. The molecular methods currently in common use also cannot determine the parental origin of each allele. Note that there may still be genetic variability present even if the detected SNPs in a region are identical in all founder individuals. Therefore, it is of interest to identify the actual origin of a region properly, even if the local marker values are of limited use for this discrimination.

1.2 Problem Description

F_2 intercross populations from outbred lines are a common experimental design [7, 8] for detection of QTL. In such designs it is non-trivial to elucidate the line of origin of QTL alleles from the genotype data. This is true even when complete and perfect marker data is available in all individuals, as the same marker allele might be present in grandparents from different lines. When the parental origin is taken into account, separating heterozygotes, an F_2 individual can have four possible genotypes at a single locus. The marker values and pedigree information can be used to evaluate which of these combinations are actually possible or admissible. E.g. if the lines are numbered 1, 2, the marker and pedigree data might show that 12, 21, 22 are possible in a specific locus in an individual, while 11 is impossible.

A multi-point algorithm for estimating QTL genotype probabilities at any point along a chromosome has been proposed for this problem [1], based on

looping over intervals of markers, each interval defined as a region between two fully informative markers. In regions between such markers, the Haldane mapping function [9] is used together with information from the partially-informative markers within the interval. Each combination of inheritance pathway realizations over all markers compatible with the data is stored, along with its probability. The resulting genotype probabilities can then be retrieved and normalized in any specific position within the interval. With some variations (e.g. in [2]), this algorithm has become the standard approach for data of this sort. Existing modifications include a pre-processing step for inference of missing marker data from other members of the pedigree, and a form of data and complexity reduction through removal of markers in dense regions.

An upper bound for the complexity of the original algorithm is $O(nk3^m)$, with n being the number of individuals, k being the total number of evaluated positions (commonly every centimorgan, so k matches the chromosome length in cM), and m the number of markers in the longest interval between fully informative markers. High m values are associated with situations where the founder individuals do not demonstrate unique marker values. Each marker imposing some constraint allows at most 3 possible inheritance pathways, hence the base value in the exponentiation.

1.3 Hidden Markov Models

As our suggested algorithms and implementations all rely on HMMs, we give a very quick review of the most critical concepts here. A general and well-written review of the subject, that introduced it to a broader audience, can be found in [10], although that article focused on language processing applications. One of the first applications in biology was in the construction of linkage maps [11], and their use has increased since.

All Markov models describe stochastic processes in some sense, and share the common property that the future evolution of the process is solely determined by the current state, hence frequently called “memory-less”. An HMM adds the concept of two processes, the “actual” Markov process proceeding between different states, and the observable process (or sequence), also called the emission process. The state in the Markov process induces a probability distribution for what symbol to emit. There exist several algorithms for HMMs, including the forward-backward algorithms for deducing state probabilities from emitted symbols, and the Baum-Welch algorithm [12] for “training” parameters in a model based on observed sequences.

2 Methods

2.1 HMMs for Genotype Probabilities

In this section, we describe the specific process of using an HMM-like approach for determining genotype probabilities, which is shared between cnF2freq and

R/qrtl. In the original description of R/qrtl [5], the benefits of using HMMs to calculate genotype probabilities were stressed. The Markov property generally holds for recombination (assuming the Haldane mapping function); the probability of a genotype switch between two positions is independent of the regions upstream or downstream. Thus, there is no need to keep the complete series of marker-value combinations per individual. The full model is then described by the state space, identifying the full inheritance pathway for each of the two alleles (4 states in the F_2 case), the transition probabilities based on the mapping function used, and the emission symbols that correspond to the observed (unordered) pairs of marker values. A detailed account of the approach for phase-known data can be found in [13].

To derive the probability of a genotype i at location x in individual k , the probability $p^{(k)}$ of the complete sequence (see below) based on marker data and expected recombination probabilities, is first computed for a given individual. This probability is the unconditional probability given by the Markov process, i.e. the probability that the observation of some arbitrary individual would match the actually observed data. Then, an additional simulated marker is added at the probing position, with the genotype to be evaluated, and the corresponding probability $p_{x,i}^{(k)}$ for this genotype is computed. The normalized probability for the genotype, given the observed sequence S , is then $p_{x,i|S}^{(k)} = p_{x,i}^{(k)}/p^{(k)}$ (Bayes law).

To compute any of the p values described above, the full process over the sequence has to be iterated. At each marker j all state probabilities $s_{j,l}^{(k)}$ are computed, where l represents the possible genotypes or inheritance pathways at that position. If the markers are located at positions x_j and the distance from the previous marker is defined by Δx_j , then the state probabilities (essentially the HMM “forward algorithm”) for a simple two-genotype (backcross) case are given by:

$$s_{j,0}^{(k)} = P_{j,0}^{(k)} \left(s_{(j-1),0}^{(k)}(1 - M(\Delta x_j)) + s_{j-1,1}^{(k)}M(\Delta x_j) \right) \quad (1)$$

$$s_{j,1}^{(k)} = P_{j,1}^{(k)} \left(s_{j-1,0}^{(k)}M(\Delta x_j) + s_{j-1,0}^{(k)}(1 - M(\Delta x_j)) \right) \quad (2)$$

Here, M is the mapping function, translating genetic distances into recombination frequencies, while P is a binary indicator for whether the state is acceptable according to the marker data. To account for the possibility of genotyping errors, the low value of P (indicating an unsupported state), can be chosen to be close to, rather than exactly 0, using a small ε . As a base case (for the first marker), we assume $s_{0,0}^{(k)} = s_{0,1}^{(k)} = 0.5$. The extension to more than the 2 states presented here is trivial, by simply adding the corresponding transition terms. The probability for a sequence is computed as the sum of all s values at the final marker.

It should be noted that the number of states should be 4 for outbred data, even for a F_2 population and related effect models, where only 3 “states” are present in the resulting linear regression (two homozygotes and one heterozygote). While the two heterozygotes should give identical phenotypes (ignoring

epigenetic effects), they are highly different from a recombination process standpoint. The transition from one heterozygote to the other actually requires two recombination events, but if those states were merged the transition would essentially be “hidden” and thus considered very likely, if compatible with the marker data. The full space of 4 states was presented in [1]. R/qlt [5] only uses 3 states when analyzing intercrosses, as heterozygote parental origin is never detectable in that case. Even for such experimental designs, though, the 3-state structure precludes the introduction of sex-specific mapping distances as the recombination probabilities relating the heterozygotes to the homozygotes become dependent on the parental origin of the alleles. Sex-specific mapping distances can easily be introduced in the 4-state model. Hence, our approach is conceptually similar to the one employed in R/qlt [5], but with applicability to the model and state space used in [1], i.e. datasets for outbred data with ≥ 4 states. With proper settings, the implemented code for our approach can give an exact reproduction of results derived with this latter existing method.

2.2 Numerical Concerns

The probabilities computed as described in section 2.1 rapidly decline, which is a general problem of HMMs [10]. The probability first computed reflects the joint probability of a specific genotype at a specific position, and the observed sequence, under the assumptions of the Markov process. This is then normalized by the probability of the observed sequence alone, with no state constraint, entailing the division of two numbers which both might be in the range of 10^{-100} or less.

In practice, we use logarithm normalization to maintain numerical accuracy, a variation of the scaling factors suggested in [10]. All calculations are performed with normal floating-point logic, but the total state vectors are at some points renormalized to sum up to 1. A logarithmized product of all normalization factors used so far is stored separately. The full vector of state probabilities essentially forms a floating-point number with a *vector-valued mantissa*, but a scalar exponent. This approach is superior to performing operations in a logarithm space, which is the method used in [5], since a slow approximation of addition is eliminated in our approach.

2.3 Multiple Interval Mapping for Linked Loci

Interval mapping can be applied to multiple loci. In this case, the references to a QTL “genotype” should be interpreted as the full configuration of alleles in all loci included. A probability for a genotype is then the probability for that full configuration. If loci are unlinked, genotypes are independent and the probability is a simple product. The same is true for linked loci separated by a fully informative marker. This can be understood by considering the probability $P(AB) = P(A) * P(B|A)$, where A and B are states describing the genotypes in two individual loci. If $P(B|A) = P(B)$, the events are independent and the product of single-locus probabilities can be used. When an informative marker

in a state C is present between the loci, $P(B|A) = P(B|C) * P(C|A)$, according to the Markov property. As C is the known state of the informative marker, $P(B|C) = P(B) * 1$.

The R/qtl package properly accounts for joint probabilities, but the implementation relies on a full sequential scan of all grid positions, resulting in $O(k^l)$ complexity, where l is the number of dimensions and k the number of grid points per dimension. Most QTL scans of high dimensionality will not consider all positions, though, but rather limit the sampling by using some optimization algorithm, like genetic algorithms [14] or branch-and-bound approaches [15], or a Markov Chain Monte Carlo approach [16]. In all these cases, many positions will never be considered. Assuming the number of queries q being relatively small compared to the total search space ($q \ll k^l$), the cost of $O(l \log m)$ per query that we present below is preferable over a full scan of $O(k^l)$.

The traditional forward-backward approach, as well as the modified multi-point approach in R/qtl rely on a linear memory representation of the search space, or a subset of it. The linearity of the space to scan does not translate to the case of two or more positions with unpredictable query ordering. The solution we propose is a hierarchical cache of “multistep transition probabilities” in a structure similar to a binary tree. The transitions from marker 0–2, 2–4, 4–6... are stored. So is 0–4, 4–8...; 0–8, 8–16..., and so on. The total memory use is still $O(nm)$ with this approach. To get the probability for specific states at a specific set of positions, the algorithm walks along the tree, using “multistep transitions” in all regions outside the markers of interest. This results in an $O(l \log m)$ factor for analyzing a specific position in a single individual.

2.4 Extension for Multiple Generations

We now present an extension of the model in [1] to more than two mating generations. As the original algorithm was exponential in the number of states, the complexity would rise from $O(3^m)$ to $O(63^m)$ by simply going from 2 to 3 generations, as we will show below. However, the complexity features of our algorithms makes this previously prohibitive extension feasible. In addition to being useful for QTL genotype estimation in multi-generational settings, the state structure described here forms the basis for the haplotype inference described in section 2.5.

The structure with 4 states only tracks a single generation of recombination, the one taking place in F_1 individuals. The state is indicating which F_0 parent contributed the allele that was subsequently transmitted to the F_2 individual. For a state to be admissible, the same marker allele should be found in the F_0 , F_1 and F_2 individuals. For each F_1 individual 2 possible alleles can be transmitted and as 2 F_1 individuals contribute to an F_2 individual, there are $2^2 = 4$ states in total. Parental origin of the F_0 alleles is unknown (“unphased data”) and thus we have no explicit state describing the F_0 individuals.

It is rather straightforward to consider an extension to 3 generations. We can model the state transitions for each F_2 individual by the 4-state space already described. There are 2 F_2 individuals contributing to an F_3 , hence $4^2 = 16$ states. Finally, 4 states are needed to describe the alleles contributed to the F_3 from

the F_2 s, or $4^3 = 64$ states in total. Every ancestor with phase information (i.e. non-founder) found in the pedigree of the individual being analyzed results in 2 additional separate states, or 1 more bit to represent the state space.

This drastic expansion from the original 4 states is caused in part by the need to track the phase for *all* F_1 individuals, even those that at the specific position of the current state are not transmitting any allele to the F_2 . If this was not done, the total number of expected recombinations would not match those observed in the data, and as a result the probabilities would be skewed and some unrealistic states could be reached.

Extension to additional generations is technically feasible, but some method of pruning (possibly “search beams” or random sampling approaches) is needed in practice, as the state space grows to $\sum_{x=1}^g 2^x = 2^{g+1} - 2$ bits, where g is the number of state-carrying generations. An extension to F_4 (3 generations with phase) would require 16384 states.

2.5 Extension for Haplotype Inference

The marker values that are measured on individuals are unphased. As described above, this limits the discriminatory power between alternative inheritance pathways in a pedigree. This problem is aggravated by the limited number of possible marker genotypes for SNP markers. When phase is unknown, the number of recombination events required to satisfy the observed HMM sequence might be reduced. This is illustrated in Figure 1, as one of the plausible interpretations when phase is removed includes inheritance pathways that would be identical in the loci shown. Here, we present an extension of the 64-state model in the previous section to include an iterative inference process for phase data in all generations, based on expectation-maximization of allele strand assignment for all individuals.

In the HMM described so far, the emitted marker value in the process was interpreted as a uniform process over all admissible symbols, i.e. marker values supported by the observed data (pedigree and marker values). Phasing can be introduced by associating non-uniform emission probabilities to the markers. These are not present in the original data, but rather need to be “trained” based on the data according to the established Baum-Welch algorithm [12]. More specifically, the markers in each individual are assigned additional “haplotype skewness” values. These values indicate the probability for the first marker value to be on strand 1 for that individual. The absolute strand numbering is arbitrary, the relevant aspect is that linked alleles should be assigned the same strand number. The assignment is initialized by fixing the first marker with non-identical alleles within each linkage group (chromosome) in every individual. The HMM probability evaluation is then repeated with shifting assumptions of whether strand 1 is maternal or paternal for the individuals in the pedigree (as no parents are available in the F_0 generation, the F_0 individuals can be excluded in this step), to make the strand numbering independent of the pedigree structure.

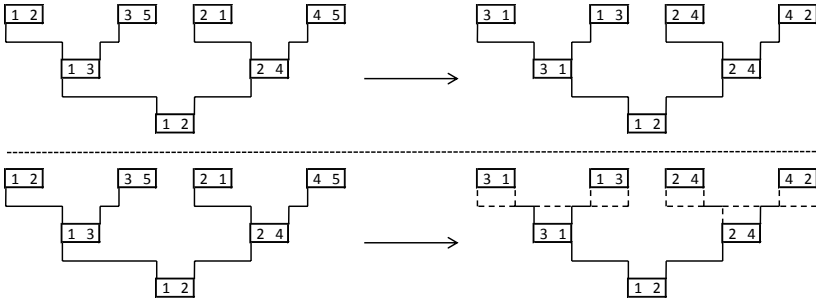


Fig. 1. The possible F_2 inheritance pathways (lines between boxes) for two closely linked loci (connected by arrows) with marker values, with (above) and without phasing (below). The marker values (boxes) are phased. The inheritance pathways indicated in the upper figure include a single recombination event, as shown by the change in the solid lines (indicating a change from maternal to paternal allele contributed from the F_1 female to the F_2 individual). In the lower figure, the alleles in the right-hand locus no longer connect to unique founders. All allowed inheritance pathways are instead indicated by dashed lines.

The need for parent-independence increases the complexity for a F_2 population by a factor of 2^3 (1 F_2 individual, 2 F_1 , all combinations should be considered). The state space used matches the one described for F_3 above, as we have 3 generations of phase information. During training, the time used for evaluation in each position is also multiplied by a factor of 2^7 , as both strands need to be considered in each individual in an analogous manner to collect frequency data, although the skewness variable will generally cause either interpretation to diminish. Already trained data (inherently one-dimensional) can then be applied to multi-dimensional queries.

The algorithm starts out with 0.5 skewness for all markers, except the ones used for defining the arbitrary strand assignment. In each iteration, the expectation for the skewness value in every marker in each individual, based on the summed probability over the states matching either of the two cases, is used to determine a new input value, to maximize the total probability. In this way, the haplotype information is extracted from the population as a whole, while single probability evaluations focus on a single individual (and the members of its pedigree). On a population level, the inherently stochastic concept of genetic linkage, expressed as the recombination events in individuals, will result in alleles that reside on the same strand cosegregating into offspring. This will favor skewness values that match the actual strand distribution of alleles. Individuals also provide conclusive phase information for different regions. As this information is included in the updated skewness value, the probabilities in all individuals that have some pedigree member with updated information become more accurate. Like most applications of expectation-maximization, the haplotypes obtained may be suboptimal. This fact can be contrasted to the other methods we present, where optimal results are indeed achieved, under the simpler models employed.

3 Results

Results from our new HMM-based implementation and the existing Fortran implementation (`ccoeff`) of the original algorithm [1] were compared for identical data from the first chromosome in an F_2 intercross bred from one Red Junglefowl male and three White Leghorn hens [17]. The dataset contains 73 genetic markers genotyped in almost 800 individuals (L. Andersson, pers comm). The total genetic map is approximately 500 cM. The marker map consists mainly of microsatellites with a high degree of variability, but also some SNPs. The presence of a few long regions of missing data in some individuals is the primary reason for the original algorithm being unable to estimate QTL genotype probabilities in the full dataset, due to the exponentially increasing memory requirement as m grows. In addition, a more typical SNP-based marker map was analyzed from a cross between Red Junglefowl and a White Leghorn chicken line with autoimmune thyroiditis (“Obese strain”, OS), similar in many respects to the first one, but with 74 markers on chromosome 1, only a handful of which are not biallelic SNPs (L. Andersson, pers comm). All lines used in the crosses are outbred, but expected to be genetically divergent. Methods for inbred lines are thus not applicable.

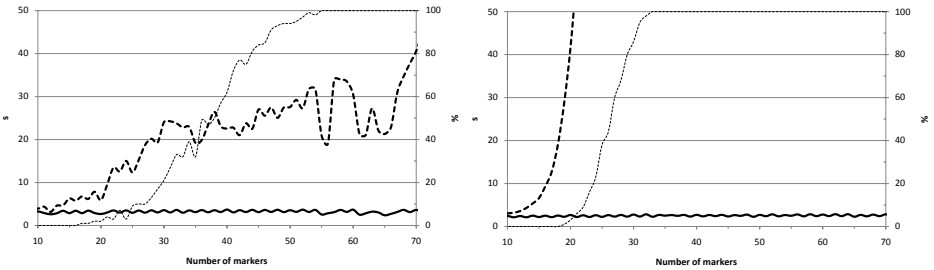


Fig. 2. Comparison of execution time obtained with `cnF2freq` (solid line) and an existing Fortran implementation (`ccoeff`; dashed) [1] for the first chromosome in two chicken intercrosses. Left: 73 markers, mainly microsatellites. Right: 74 markers, all but a handful being biallelic SNPs. Randomly sampled subsets of full marker set used in each run, 100 runs per marker count. Thin dashed line indicates the proportion of `ccoeff` terminating early due to running out of memory space, also explaining the time-use plateau in the microsatellite case. The binary tree algorithm results in alternating behavior for odd and even total marker counts.

The time used on a 4-core (2-socket) machine was tested with 100 random subsets of the total marker map for different marker counts, from 3 to 40. Both algorithms were exposed to identical random samples. As the HMM algorithms are not limited by memory and shows significant locality, `cnF2freq` has been parallelized by using OpenMP. Probabilities were evaluated every 1 cM. The flanking markers on the chromosome were always included, to keep the total length consistent over all runs. The wall-clock time use is presented in Figure 2.

In the graphs there is also a secondary (right-hand) scale illustrating the proportion of random samples that were terminated early in `ccoeff`, due to the number of search paths exceeding 2^{22} , essentially filling a 32-bit address space. In both datasets, the full results from `cnF2freq` are always determined in under 4 seconds, while the time and memory usage grow exponentially in `ccoeff`. The higher number of SNPs in the second cross results in more drastic growth. In all successfully completed runs, the results from the two methods were identical, save some differences in rounding.

The haplotyping functionality was tested on the common simulated dataset from the QTLMAS XII [18] conference. This is a multigenerational dataset with a very high marker density. Thanks to its simulated origin, phased data is available. In total, about 6000 individuals were fitted against an unphased version of the dataset, with 1000 markers over the first chromosome, 100 cM in length. 99.0% of heterozygous markers were correctly phased over all individuals where genotypes were provided. Our results thus provide more accurate haplotype estimates than those presented at the conference, based on the Minimum Recombination Haplotype Configuration (MRHC) [19] method. This method produced a precision of 98.6% on a subset of 95.3% of heterozygous markers, with 4.7% excluded as unassignable, resulting in a total precision of 94.0%.

4 Discussion

The most basic and immediate use for this tool is a drop-in replacement for existing code used to estimate QTL genotype probabilities in F_2 intercrosses from outbred founders. Like the approach and related code described in [1], we are able to handle missing marker data and marker ambiguity in outbred F_2 populations. We have shown that we avoid the exponential increase in memory use and run time that is a limitation for these methods when experimental marker maps used in QTL mapping experiments are changing from 10s or 100s of highly informative multi-allelic microsatellites to 1000s of less informative bi-allelic SNPs.

The web-based QTL Express [2] tool (now replaced by GridQTL, <http://www.gridqtl.org>) implemented the original exponential algorithm, but with some improvements for inference of some missing marker values, and a scheme to reduce the value of the m component in the complexity calculations. This is done with a loss of precision, however. The markers in a not completely informative region for an individual are iteratively reduced, removing at each step the single marker that will leave the shortest gap, i.e. where the two surrounding markers are closest together. With SNP maps, considerable information can be inferred from preserved clusters of highly linked markers, where the combination only leaves one possibility with non-minimal probability. When the marker set in ambiguous regions is “thinned out” to about 15 markers in total (the maximum number feasible with an exponential approach), such clusters are the first to be eliminated, and informativeness eventually is lost everywhere.

4.1 Haplotype Inference

Starting from the foundation of reimplementing existing models for genotype inference from multipoint marker data, we have also shown how other established aspects of Hidden Markov Model theory can be adapted to solve the haplotype inference problem, either in its own right, or as a tool to improve the genotype estimates. We have demonstrated the accuracy of this haplotyping algorithm on a publicly available simulated dataset. It is worth noting that in the cases where our algorithm failed on the simulated dataset, the sequences on both sides were highly homologous, making a unique haplotype assignment almost impossible.

There are several existing methods for determining haplotypes from unphased data in different pedigrees and marker configurations. Most of the research in the area, like application of EM [20], Clark's algorithm [21] and Bayesian methods [22] were designed for cases where no pedigrees are available, and in some cases also with an assumption of Hardy-Weinberg equilibrium. Some approaches based on MCMC (Markov Chain Monte Carlo) methods can integrate general pedigree information, but do so by sampling specific realizations and then analyzing them.

The MRHC approach, to which we have compared our results, does include pedigrees. This approach is a rule-based application of the principle that a haplotype resolution over the pedigree should minimize the number of recombinations. Our approach demonstrates a conceptual similarity as the presence of recombination events reduces the probability. Although only 3 generations can be feasibly maintained in an individual analysis, pedigrees of more generations can be treated by breaking them into smaller analyses of 3 generations at a time. If the phase of some region is undetermined the skewness is simply neutral, allowing for both interpretations. When the phase information for a locus is assigned, there is no need for further ancestral information. Close to the point of convergence, the haplotyping problem is inherently local. This is also demonstrated in our experiments, as we could solve the haplotypes in up to 7 generations with great success, despite each analysis being constrained to 3 generations.

5 Conclusion

We have demonstrated the ability of `cnF2freq` to be used both for providing results identical to the ones obtained with established methods, and to improve accuracy by a model reconstructing phased data. In both cases, we observe significant time savings, allowing efficient use on current, dense marker maps for outbred F_2 data. It is clear that a more thorough use of HMM-based methods will add considerable value to the analysis of current and future SNP-based maps with 1000s of markers. The current code is implemented in C++ and is simple to integrate into existing systems, as it is relatively platform-independent. Work is underway to utilize this technology in an implementation of the multi-dimensional search approach described in [15].

Acknowledgements

This work has been funded by The Graduate School in Mathematics and Computing (FMB), Sweden. The work by ÖC was funded by Swedish Foundation for Strategic Research. Per Jensen, Leif Andersson, and Olle Kämpe are acknowledged for sharing experimental data for evaluation of the method.

References

1. Haley, C.S., Knott, S.A., Elsen, J.M.: Mapping Quantitative Trait Loci in Crosses Between Outbred Lines Using Least Squares. *Genetics* 136(3), 1195–1207 (1994)
2. Seaton, G., Haley, C.S., Knott, S.A., Kearsley, M., Visscher, P.M.: QTL Express: mapping quantitative trait loci in simple and complex pedigrees. *Bioinformatics* 18(2), 339–340 (2002)
3. Doerge, R.W.: Mapping and analysis of quantitative trait loci in experimental populations. *Nat. Rev. Genet.* 3(1), 43–52 (2002)
4. Lander, E.S., Botstein, D.: Mapping Mendelian Factors Underlying Quantitative Traits Using RFLP Linkage Maps. *Genetics* 121(1), 185–199 (1989)
5. Broman, K.W., Wu, H., Sen, S., Churchill, G.A.: R/qtll: QTL mapping in experimental crosses. *Bioinformatics* 19(7), 889–890 (2003)
6. Vignal, A., Milan, D., SanCristobal, M., Eggen, A.: A review on SNP and other types of molecular markers and their use in animal genetics. *Genet. Sel. Evol.* 34(3), 275–305 (2002)
7. Slate, J.: Quantitative trait locus mapping in natural populations: progress, caveats and future directions. *Molecular Ecology* 14(2), 363–379 (2005)
8. Andersson, L., Georges, M.: Domestic-animal genomics: deciphering the genetics of complex traits. *Nat. Rev. Genet.* 5(3), 202–212 (2004)
9. Haldane, J.B.S.: The combination of linkage values, and the calculation of distances between the loci of linked factors. *Journal of Genetics* 8, 299–309 (1919)
10. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2), 257–286 (1989)
11. Lander, E.S., Green, P.: Construction of multilocus genetic linkage maps in humans. *Proc. Natl. Acad. Sci. U S A* 84(8), 2363–2367 (1987)
12. Baum, L.E., Petrie, T., Soules, G., Weiss, N.: A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics* 41(1), 164–171 (1970)
13. Broman, K.W.: Use of Hidden Markov Models for QTL mapping. Working Paper 125, John Hopkins University, Dept. of Biostatistics (2006)
14. Carlborg, O., Andersson, L., Kinghorn, B.: The Use of a Genetic Algorithm for Simultaneous Mapping of Multiple Interacting Quantitative Trait Loci. *Genetics* 155(4), 2003–2010 (2000)
15. Ljungberg, K., Holmgren, S., Carlborg, O.: Simultaneous search for multiple QTL using the global optimization algorithm DIRECT. *Bioinformatics* 20(12), 1887–1895 (2004)
16. Sillanpaa, M.J., Arjas, E.: Bayesian Mapping of Multiple Quantitative Trait Loci From Incomplete Inbred Line Cross Data. *Genetics* 148(3), 1373–1388 (1998)
17. Kerje, S., Carlborg, O., Schütz, K., Hartmann, C., Jensen, P., Andersson, L.: The twofold difference in adult size between the red junglefowl and White Leghorn chickens is largely explained by a limited number of QTLs. *Anim. Genet.* 34(4), 264–274 (2003)

18. Crooks, L., Sahana, G., de Koning, D.J., Sando Lund, M., Carlborg, O.: Comparison of analyses of the QTLMAS XII common data set II: genome-wide association and fine mapping (submitted) (2008)
19. Li, J., Jiang, T.: Efficient inference of haplotypes from genotypes on a pedigree. *J. Bioinformatics and Computational Biology* 1(1), 41–70 (2003)
20. Excoffier, L., Slatkin, M.: Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Mol. Biol. Evol.* 12(5), 921–927 (1995)
21. Clark, A.: Inference of haplotypes from PCR-amplified samples of diploid populations. *Mol. Biol. Evol.* 7(2), 111–122 (1990)
22. Niu, T., Qin, Z.S., Xu, X., Liu, J.S.: Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *Am. J. Hum. Genet.* 70(1), 157–169 (2002)

A Comprehensive Analysis Workflow for Genome-Wide Screening Data from ChIP-Sequencing Experiments

Hatice Gulcin Ozer^{1,2}, Doruk Bozdağ^{1,3}, Terry Camerlengo¹, Jiejun Wu⁴,
Yi-Wen Huang⁴, Tim Hartley¹, Jeffrey D. Parvin^{1,2}, Tim Huang⁴,
Umit V. Catalyurek¹, and Kun Huang^{1,2}

¹ Department of Biomedical Informatics, The Ohio State University

² The Ohio State University Comprehensive Cancer Center Biomedical Informatics Shared Resource

³ Department of Electrical & Computer Engineering, The Ohio State University

⁴ Department of Molecular Virology, Immunology, The Ohio State University
43210 Columbus, OH, USA

{ozer, khuang}@bmi.osu.edu

Abstract. ChIP-sequencing is a new technique for generating short DNA sequences useful in analyzing DNA-protein interactions and carrying out genome-wide studies. Although there are some studies to process and analyze ChIP-sequencing data, a complete workflow has not been reported yet. The size of the data and broad range of biological questions are the main challenges to establish a data analysis workflow for ChIP-sequencing data. In this paper, we present the ChIP-sequencing data analysis workflow that we developed at the Ohio State University Comprehensive Cancer Center Bioinformatics Shared Resources. This pipeline utilizes 1) use of different mapping algorithms such as Eland, MapReads, SeqMap, RMAP to align short sequence reads to the reference genome 2) a novel normalization algorithm to detect significant binding densities and to compare binding densities of different experiments 3) gene database mapping and 3D binding density visualization 4) distributed computing and high performance computing (HPC) support.

Keywords: ChIP-seq, workflow, short sequence mapping, parallelization, normalization, visualization.

1 Introduction

Massive parallel sequencing is a high-throughput technology which can sequence tens of millions of DNA segments in a single experiment. It has been widely used in many genome-wide studies such as microRNA screening, genome re-sequencing, and protein-DNA interaction (ChIP-seq). For the latter, it has demonstrated significant advantage over the ChIP-chip approach with high resolution, comprehensive coverage, and low cost [1,2]. Despite the sharply increasing demand in using ChIP-seq technology and analyzing data from such experiments, currently there are very few pieces of work on data analysis methods and there is no complete workflow outlined for this type of experiments.

Here, we present the ongoing work on developing a comprehensive data analysis workflow for ChIP-seq experiments at the Ohio State University Comprehensive Cancer Center Biomedical Informatics Shared Resources (OSUCCC BISIR). The development of the workflow includes the following key steps:

1. Integrating use of different mapping algorithms such as ELAND (from Illumina) [3], MapReads (from Applied Biosystems) [4], SeqMap [5], RMAP [6] and xMAN [7] to align short sequence reads from ChIP-seq experiments.
2. Developing new algorithms including detecting genome segment detection with significant high density of protein binding and a new global normalization algorithm for comparing binding density between different experiments.
3. Developing software infrastructure for binding segment visualization, gene database mapping, novel 3-D binding density landscape rendering, and pathway analysis.
4. Integrating the workflow components into a distributed computing environment with high performance computing (HPC) support for key steps such as mapping the sequenced DNA segments to the genome [8].
5. Integrating the workflow into our data management system, QUEST (QUERy Support Tool for epigenetics).

2 Methods: Chip-Seq Analysis Workflow

Fig. 1 demonstrates our workflow for analyzing ChIP-seq data. It contains three major components: data preprocessing, analysis/visualization, and data management.

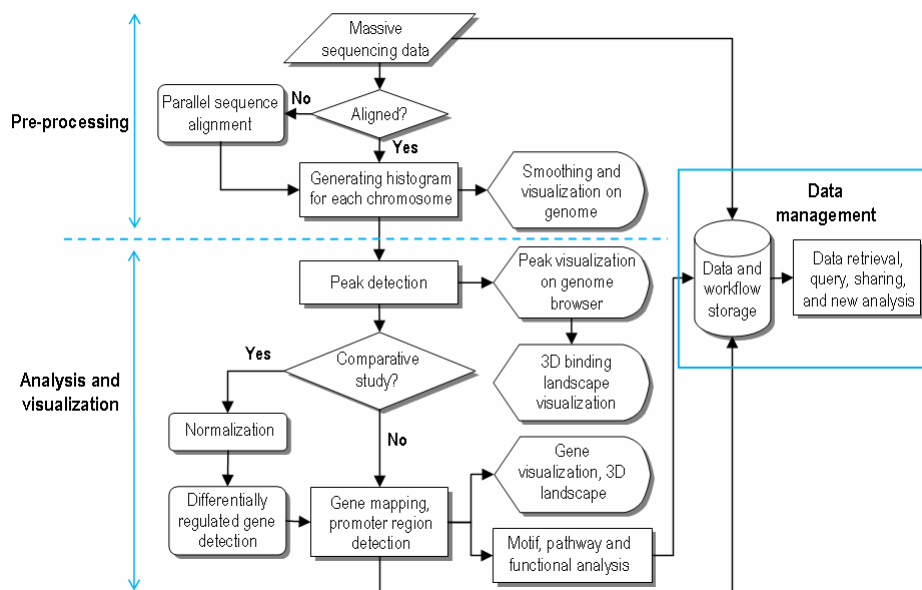


Fig. 1. The workflow for ChIP-seq data processing and analysis

Data preprocessing component includes mapping sequences to reference genome using multiple software tools, parallel implementation of mapping algorithms, generation of standard output files that can be visualized in UCSC Genome Browser, and mapping of DNA fragments to RefSeq gene database. Analysis component consists of detection of high density binding regions, normalization of the samples for comparative analysis, and 3D binding landscape visualization. Data management component utilizes data storage and access.

2.1 Preprocessing

2.1.1 Mapping Sequences to Reference Genome

The ChIP-seq data generated from sequencing equipments such as the Solexa and SOLiD sequencers are in the form of short DNA segments of no more than 50 bases. These short DNA segments need to be aligned to the reference genome before further processing. While the commercial vendors usually provide sequence alignment software such Eland provided by Illumina as part of the Solexa data analysis pipeline, they are usually not sufficient for all situations.

For instance, Eland is hard coded to align sequences up to 32 nucleotides in length and allow at most 2 mismatches. In some cases it is necessary to use full length of the sequence segments and allow more than 2 mismatches, even gaps, for the alignment. To overcome limitation of Eland [5], we also integrate several other algorithms including SeqMap, RMAP, MapReads, and xMAN into our workflow. The SeqMap algorithm allows up to 5 mismatches and gaps in combination and use full length of the sequence segments. In addition, at the end of Solexa pipeline, base-call quality scores for the sequence segments are reported. Eland uses these quality scores to filter out low quality sequence reads. These quality scores can be integrated into the alignment process by evaluating only the high quality bases in the sequence segments during the alignment. This approach is implemented in RMAP algorithm. Compared to Eland and SeqMAP, RMAP is much slower [5]. However, RMAP provides better mapping accuracy [6]. Therefore, in addition to Eland and SeqMap, we also integrate RMAP algorithm into our workflow.

2.1.2 Parallel Implementation of the Sequence Mapping Algorithms

The preprocessing step is the most computationally intensive part of the workflow. Current second generation sequencers generates up to billion bases per date, and a single run could generate more than hundred million reads. Mapping those millions of reads could take hours to days if the mapping is computed sequentially. Therefore we also integrate novel techniques that allow distributing the mapping computations to multiple compute nodes [8].

In our workflow we consider two parallelization approaches. In the first approach, we use DataCutter, a middleware we developed previously for staging large datasets to multiple nodes in a computer cluster [9], to wrap and parallelize a mapping algorithm by distributing computation involving parts of the reference genome on multiple nodes of a cluster. In the second approach, we developed novel parallelization methods to distribute the work involving both the reference genome and the sequences to be mapped to the genome in an efficient way [8]. These methods are designed to optimize the distribution of data to enhance the parallel performance and can be used to parallelize most mapping algorithms.

2.1.3 Output Files and Initial Visualization

The sequence mapping results are stored in files containing millions of chromosomal locations and strand orientation tags. Such files cannot be interpreted directly by biologists. In order to facilitate the interpretation, our workflow automatically splits the sequencing mapping results based on the chromosomes, i.e. the segments mapped to each chromosome stored in a separate file. In addition, for each chromosome, the file is converted to the BED and WIG formats. BED and WIG files are the common file formats used by the UCSC Genome Browser. The BED files allow user to visualize the binding locations of sequence tags and the WIG files allows user to visualize computed binding densities of the sequence tags over the genome.

In addition, the workflow also contains a module to automatically generate the histogram of the density of the sequence tags. Specifically, the workflow generates the histogram for each chromosome using three different bin sizes: 200bp, 500bp, 1000bp. The choice of the bin sizes is based on the fact that most of the DNA segments generated in the ChIP experiments are between 200-500bp. The histogram files are essential for the next stage of data processing – the analysis and visualization stage.

2.1.4 Gene Mapping

DNA fragments (bins) can be mapped to a local copy of the UCSC Genome Browser with databases such as the RefSeq gene database using a Java script. In addition, by entering a gene name, the visualization tool can display the histogram of sequence tag density of the gene including both its upstream and downstream 2kb flanking regions.

2.2 Analysis

The mapped segments are then sorted based on their locations in the chromosomes and saved in files of BED and WIG formats, which can be visualized as custom tracks in the UCSC Genome Browser. Furthermore, the histograms with bin size of 1kb are generated for every chromosome. For each bin, we record the number of DNA segments it contains.

2.2.1 A Poisson Model for Detecting High Density Regions

We developed a Poisson model for detecting bins with significant high binding density. For each chromosome, we compute the parameter λ of the Poisson distribution using the total amount of aligned segments and total number of bins for this chromosome. This λ is then used to compute the odds of binding density for each bin which is further converted into a p-value. The bins are then ranked by the significance p-values for selection and peak detection. A train of consecutive bins with significantly high binding densities can be combined into a large fragment.

2.2.2 A Global Normalization Method for Comparative Analysis

In a comparative study, two sets of data needs to be compared in order to identify regions for which the protein interaction is differentially regulated. The key step for a comparative study is to normalize the two sets of data in order to obtain meaningful comparison results. We developed a global normalization method which is similar to the linear normalization algorithm as summarized below for two-color genechips [10].

Algorithm 1. Global normalization of ChIP-seq data between two samples.

1. For the i th chromosome in the k th sample ($k = 1, 2$), generate the histogram H^k of binding density over the genomic locations with bin size h (here we set $h = 1000$ and 5000 depending on the application). We set sample 1 as the reference sample.
2. Compute the total sequence tags T_i^k for the i th chromosome in the k th sample ($k = 1, 2$).
3. Compute the ratio $r = T_i^2/T_i^1$.
4. For each bin in the histogram of sample 2, multiply r .
5. Visualization: Plotted histograms of sample 1 against sample 2 as scattering plot for each chromosome. Then, we used MA plots to visualize similarity of the two samples in binding densities. The X-axis represents the difference in average log intensities ($M = \log_2 H^1 - \log_2 H^2$); whereas the Y-axis represents the log-ratio ($A = (\log_2 H^1 + \log_2 H^2)/2$). The normalization process moved the median to zero in MA-plots to achieve global normalization.

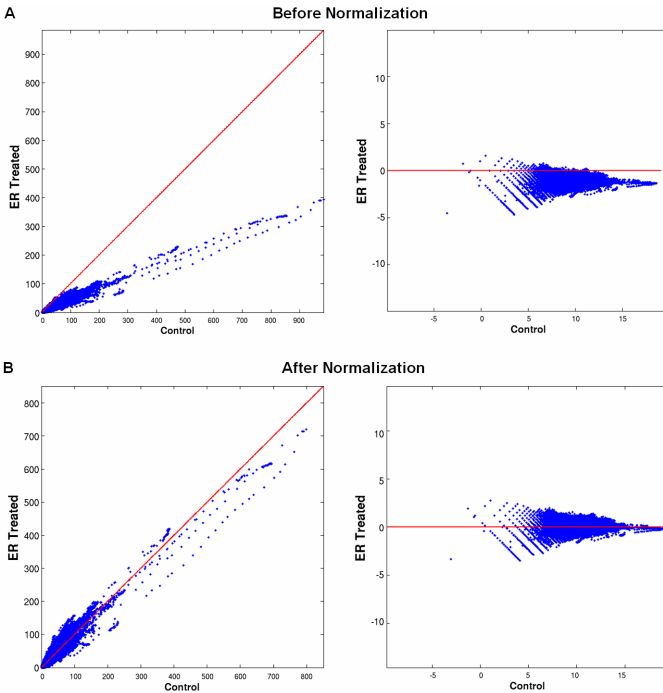


Fig. 2. Data normalization. The binding densities for corresponding regions of 20kb long with the OHT control sample being the x-axis and the OHT E2 treated sample being the y-axis, before (A) and after (B) normalization. The right plots are the MA plots with x-axis being the ratio between the two samples after logarithm transformation with base 2. The y-axis is the product between the two samples.

Fig. 2 shows the the MA-plot before and after the normalization in one of our studies for human chromosome 10. Given the cost of the ChIP-seq experiments, duplicates in experiments are rare and therefore differential regulated regions are detected using threshold on the fold change.

DNA fragments (bins) selected in the above steps are then mapped to the RefSeq gene database using the Java script described in Section 2.1.4. Fragments within 2kb from the transcription starting sites of genes or having overlaps with genes transcription regions are reported for further analysis. The pathway and functional analysis of the genes are conducted using the Ingenuity Pathway Analysis (IPA) software (<http://www.ingenuity.com>) and other ontology enrichment analysis tools such as BiNGO (<http://www.psb.ugent.be/cbd/papers/BiNGO/>).

2.3 Data Visualization

We have developed a multi-resolution visualization tool name GenomeScape in Matlab which allows us to visualize the ChIP-seq data at multiple scale of chromosomal resolution. The approach is analogous to that commonly used in Google maps (<http://maps.google.com>) which starts with a view of the mainland United States and can zoom in to view at the resolution for a single state, city, or neighborhood. Similarly, when viewing the ChIP-seq results, the whole genome view is informative for some experiments, whereas large chromosomal domains and gene clusters provide a useful view and single gene resolution is also valuable. For the full genome view, we array each chromosome in the X-Y plane, and the histogram for sequence tags is depicted in the Z-axis (see Fig. 7).

2.4 Data and Workflow Management

We developed a data management and workflow control system named QUEST. This system allows the user to store, share, query, and retrieve large bioinformatics data including genechip and ChIP-sequencing. In addition, it enables the user to carry out pre-defined data analysis workflow in many scripts including Matlab, R, Perl, Python and Java. Currently the preprocessing workflow, generating BED files, WIG files and histogram files, for the mapping results are integrated into the QUEST system.

3 Results

3.1 Parallelization of Sequence Mapping Algorithms

Even though the implementation is not fully optimized, our first parallelization approach using DataCutter yielded fairly good performance for medium sized datasets. For a test with 13 million 36bp sequences, we were able to reduce the computing time by 7.5 fold using eight nodes. The matching time was reduced from 75 hours (4500 minutes) to 10 hours (600 minutes).

In a separate test, using a computer cluster with 32 nodes, we were able to map two datasets with 6.6 million 36bp sequence tags and 25 million sequence tags to the

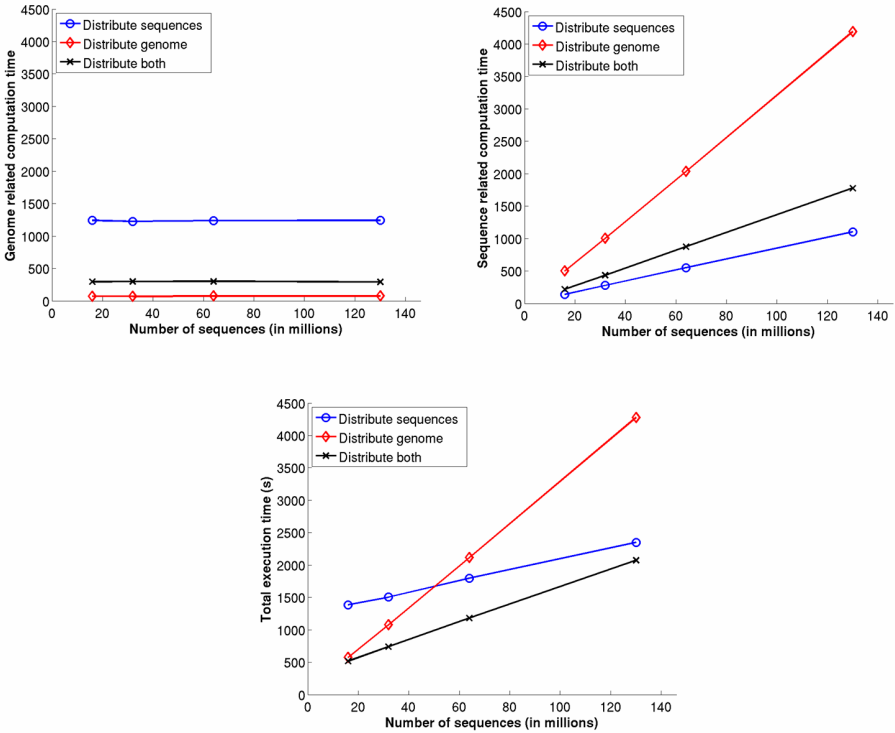


Fig. 3. Comparison of three parallelization methods while mapping different numbers of sequences to a reference genome of 800bp length using parallelized MapReads algorithm on a 16-node cluster. Efficient distribution of both sequence and genome data improves the performance compared to the cases where only either sequence or genome data is distributed.

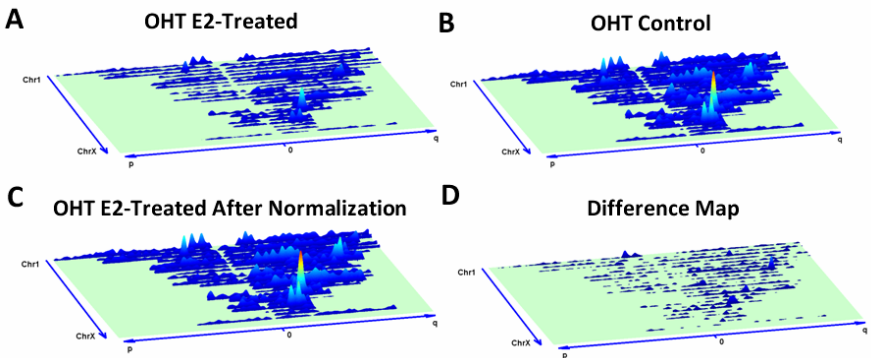


Fig. 4. Data normalization and comparison. A) The 3-D landscape of the binding density of OHT E2 treated sample before normalization. B) The 3-D landscape of the binding density of OHT control. C) The 3-D landscape of the binding density of OHT E2 treated sample after normalization. D) The 3-D difference map between OHT E2 treated sample and the OHT control sample.

entire human genome in 33 minutes and 89 minutes respectively. For this test we applied our parallelization methods [8] to the MapReads algorithm for distributing both sequences and the genome and allowed up to three mismatches. As demonstrated in Figure 3, efficient distribution of sequence and genome data helps improving the parallel runtime of an algorithm.

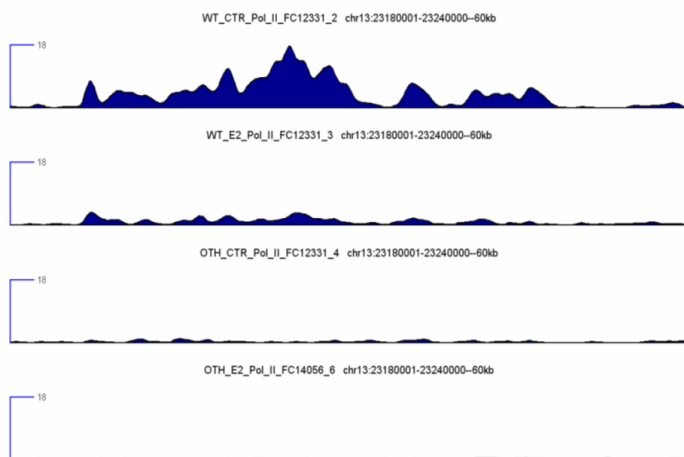


Fig. 5. An example of a 60kb segment on chromosome 13 in which the responses to E2 are different between the two cell stains. The four lanes of histograms represent data from the MCF7 cell control, the MCF7 cells treated with E2, the OHT cell control, and the OHT cells treated with E2.

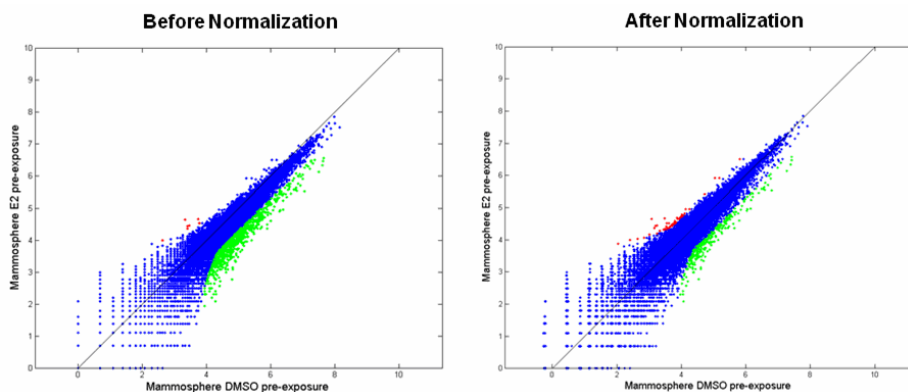


Fig. 6. The binding density for more than 18,000 genes for the estrogen treated sample versus the control sample is plotted in the log-log plot

3.2 Application of the Workflow on Studying Drug Resistance in Breast Cancer

The motivating biomedical application for us to develop this system is a study on the responses to estrogen (E2) for two different cell lines: MCF7 – a commonly used breast cancer cell line, and OHT – a breast cancer cell strain that is resistant to the anti-cancer drug tamoxifen. In this study we used CHIP-seq experiments to screen for regions with Polymerase II (Pol II) binding in order to characterize the effects of E2 treatment on gene transcription to understand the difference between the tamoxifen-resistant strain and regular cancer cell strain.

We carried out the global normalization between E2 treated samples and the corresponding controls. As shown in Fig. 4, the global normalization corrected the scale difference between corresponding samples. The difference map (Fig. 4D) shows that

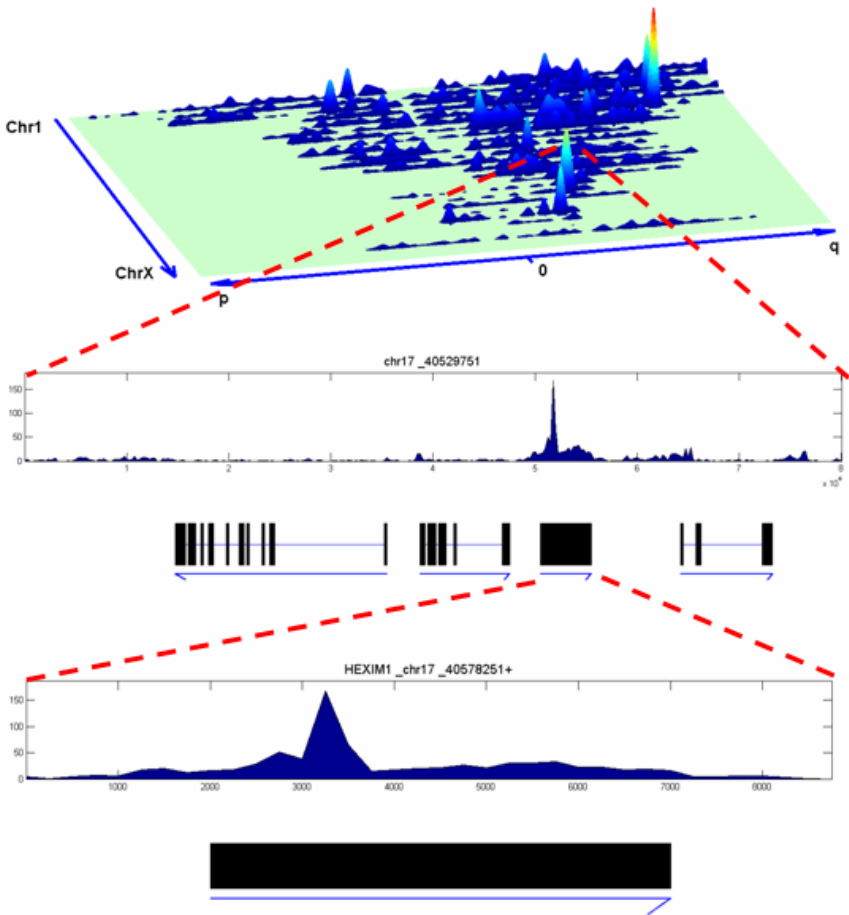


Fig. 7. A multi-resolution visualization over the genome. Top: the RNAPII binding landscape over the entire genome. Middle: the gene cluster (four genes with exons marked by black boxes and directions of the genes marked by the arrows) around the peak on chromosome 17. Bottom: the binding pattern over a gene (HEXIM1).

in the OHT cell strain, there are many regions with small responses but few regions with large responses to E2 treatment. We further detected 465 regions with significant difference in responses to E2 between the two cell lines with a total length of 39.8 millions of bases. An example is shown Fig. 5. In this example, the binding density of Pol II was significantly reduced after the E2 treatment in MCF 7 cells while there was no response to E2 for the OHT cells.

In addition to the chromosomal regions (individual bins in the histogram), we also applied the global normalization algorithm to the genes. As shown in Fig. 6, the scatter plots for the more than 18,000 genes in the RefSeq databases are generated between the E2 treated samples and the control samples. The distribution is apparently biased before normalization (Fig. 6 left) but was corrected after normalization (Fig. 6 right). This step allows us to identify 372 up-regulated genes and 611 down-regulated genes with at least 1.5 fold changes after E2 treatment in terms of Pol II binding densities.

3.3 Multiscale Visualization of the ChIP-seq Data

The software GenomeScope allows us to navigate through the genome to study the binding patterns of the proteins on the genome at different resolutions. As shown in Fig. 7, the landscape of the Pol II binding density histogram over the entire genome for the MCF7 cells treated with E2 is generated and when we zoom in onto one of the highest peaks we identified a gene named HEXIM1 which codes for protein that is a general Pol II inhibitor. This interesting observation raises further biological question on if this is there is a negative feedback mechanism for Pol II activity.

4 Conclusion and Discussion

We presented a comprehensive data analysis workflow for ChIP-seq experiments. With all the key steps being worked out, we plan to integrate these steps so that the data can be analyzed in an automatic or semi-automatic fashion. While the system is developed for analyzing ChIP-seq data, it can be adapted to process other high throughput sequencing data such as microRNA screening data.

Currently we are working on two fronts to improve this system: developing new algorithms and integrating the softwares. We are currently developing a more sophisticated nonlinear data normalization algorithm based on lowess fitting for comparative experiment data. In addition, we are developing new statistical models for more accurate peak detection as well as a wavelet-based algorithm for multi-resolution enriched region discovery. In the software development front, currently our focus is to integrate the individual pieces of the workflow including the mapping algorithms, data visualization and some routine analysis algorithms (e.g., peak identification) into the QUEST system.

Workflow tools are available upon request.

References

1. Johnson, D.S., Mortazavi, A., Myers, R.M., Wold, B.: Genome-wide mapping of in vivo protein-DNA interactions. *Science* 316(5830), 1497–1502 (2007)
2. Robertson, G., Hirst, M., Bainbridge, M., Bilenky, M., Zhao, Y., Zeng, T., Euskirchen, G., Bernier, B., Varhol, R., Delaney, A., Thiessen, N., Griffith, O.L., He, A., Marra, M., Snyder, M., Jones, S.: Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nature Methods* 4, 651–657 (2007)
3. Cox, A.: ELAND: Efficient Local Alignment of Nucleotide Data (unpublished)
4. Zhang, Z., et al.: Fast flexible mapping of AB SOLiD short sequence reads (unpublished)
5. Jiang, H., Wong, W.H.: SeqMap: mapping massive amount of oligonucleotides to the Genome. *Bioinformatics* 24(20), 2395–2396 (2008)
6. Smith, A.D., Xuan, Z., Zhang, M.Q.: Using quality scores and longer reads improves accuracy of Solexa read mapping. *BMC Bioinformatics* 9, 128 (2008)
7. Li, W., Carroll, J.S., Brown, M., Liu, S.: xMAN: extreme MAPPING of OligoNucleotides. *BMC Genomics* 9(suppl. 1), S20 (2008)
8. Bozdag, D., Barbacioru, C., Catalyurek, U.: Parallel Short Sequence Mapping for High Throughput Genome Sequencing. In: 23rd International Parallel and Distributed Processing Symposium (to appear) (2009)
9. Beynon, M.D., Kurc, T., Catalyurek, U., Chang, C., Sussman, A., Saltz, J.: Distributed processing of very large datasets with DataCutter. *Parallel Computing* 27(11), 1457–1478 (2001)
10. Dudoit, S., Yang, Y.H., Callow, M.J., Speed, T.P.: Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica Sinica* 12, 111–140 (2002)

A Fitness Distance Correlation Measure for Evolutionary Trees

Hyun Jung Park^{1,*} and Tiffani L. Williams²

¹ Department of Computer Science, Rice University
hp6@cs.rice.edu

² Department of Computer Science and Engineering, Texas A&M University
tlw@cse.tamu.edu

Abstract. Phylogenetics is concerned with inferring the genealogical relationships between a group of organisms (or taxa), and this relationship is usually expressed as an evolutionary tree. However, inferring the phylogenetic tree is not a trivial task since it is impossible to know the true evolutionary history for a set of organisms. As a result, most phylogenetic analyses rely on effective heuristics for obtaining accurate trees. These heuristics use tree score as a basis for establishing an accurate depiction of evolutionary tree relationships. Relatively little work has been done to analyze the relationship between improving tree scores (fitness) and topological accuracy (distance). In this paper, we present a new fitness-distance correlation coefficient called r_{FD} to quantify the relationship between evolutionary trees. By applying this measure to three biological datasets consisting of 44, 60, and 174 taxa, our results show that improvements in fitness are strongly correlated ($r_{FD} > 0.8$) with topological accuracy to the *best-tree-overall*. Moreover, we investigated the use of the r_{FD} coefficient if the best overall tree is not available and found similar results. Thus, our results show that r_{FD} is a robust measure with several potential applications such as the development of stopping criteria for phylogenetic search.

1 Introduction

Given a collection of n organisms (or taxa), the objective of phylogeny reconstruction is to produce a phylogenetic tree describing the evolutionary relationships between the organisms. Evolutionary trees attempt to predict the past with information (e.g., biomolecular sequences, morphological data) from current-day organisms. From tracking the transmissions of diseases to improving agricultural practices to meeting the needs of a growing population, the societal impact of phylogeny reconstruction is tremendous. Maximum parsimony (MP) and maximum likelihood (ML) are two of the major optimization problems used to reconstruct phylogeny reconstruction, but both are quite difficult to solve since they are NP-hard problems. Hence, heuristics are used to search for the optimal-scoring tree in the exponentially-sized tree space. (For n taxa, there are

* Most of this work was done while at Texas A&M University.

($2n-5$)!! possible evolutionary trees.) However, the actual objective of phylogeny reconstruction is to obtain a reasonable estimate of the underlying evolutionary history (as represented by the branching order, or “topology” of the phylogeny). We investigate whether the goals of a phylogenetic heuristic (i.e., finding the optimal-scoring tree) correspond to the actual goal of phylogenetics, which is depicting accurate relationships between organisms (i.e., topological accuracy).

The optimization function value is the primary guidance mechanism that is used by phylogenetic heuristics in their search for a (globally optimal) solution to a given dataset. For this guidance to be effective, ideally, the better the trees are rated, the closer they are to the true evolutionary history. In this paper, we evaluate the nature of the relationship between the tree scores and the distance between trees for a given search landscape. We develop a *new measure*, which was motivated by the work of Jones and Forrest for genetic algorithms [10], to summarize the relationship between tree fitness and tree distance. Our measure, called r_{FD} , tests whether better-scoring trees correspond to more accurate topologies. Hence, r_{FD} measures the correlation between fitness (tree scores) and distance (topological accuracy) to a target tree of interest. (The term fitness is often used to refer to evaluation or objective function values.) Given that r_{FD} is a correlation coefficient, its value ranges from -1 to +1.

Our contributions. Our investigation of the correlation between fitness and distance addresses the following research questions.

1. What is the correlation between tree scores and their topological distance to the *best-tree-overall*?
2. How sensitive is the r_{FD} measure to having access to the *best-tree-overall*? In other words, what is the robustness of the r_{FD} measure if one uses the *best-tree-so-far*?

Given that the true evolutionary history for a set of organisms is unknown, a reasonable substitute is to use the best-scoring tree found by any phylogenetic method as the *best-tree-overall* or “true” tree. The other target tree of interest is the *best-tree-so-far*. A phylogenetic heuristic may not always have access to the best overall tree—especially if the dataset of interest has been newly created. However, every heuristic will have access to its *best-tree-so-far*, which changes as the phylogenetic search makes improving moves based on fitness in its attempt to find the optimally-scoring tree.

To address the above questions, we developed a heuristic called *Simple Local Search (SLS)* that reconstructs trees based on the maximum parsimony criterion. Our SLS heuristic is an implementation of the hill-climbing strategies employed in popular phylogenetic software packages. Since we implemented SLS, we can collect a variety of data regarding the choices that SLS makes during a search. One of the advantages of using SLS is that we can collect trees of all different fitness values on its search for the best-scoring tree. Commercial phylogenetic software such as PAUP* [17] does not provide us with this type of control. Also, we were unable to get TNT [9] to provide us with this type of functionality. Although SLS performs comparably to PAUP* and outperforms Phylip [3], our

goal is not to recommend that SLS replace other established phylogenetic search methods. We simply designed SLS to collect all the phylogenetic trees it visits in tree space in order to explore the correlation between fitness and distance between a collection of evolutionary trees.

With SLS, we study the fitness distance correlation coefficient, r_{FD} on three biological datasets consisting of 44, 60, and 174 taxa. Our results show that improvements in fitness (MP scores) are strongly correlated ($r_{FD} > 0.8$) with topological distance to the *best-overall-tree*. For random trees, the r_{FD} value ranges from 0.2 to 0.4. The r_{FD} correlation coefficient also shows high correlation when the search only has access to the *best-tree-so-far*. One of the interesting features of the r_{FD} value in this context is that as the search progresses through tree space the r_{FD} values increase. Hence, the r_{FD} values near the end of a search are much higher than those during the early stages of the search. Hence, the r_{FD} value can be used as a mechanism for monitoring the amount of diversity found in the trees found during the search. It can also be use as a way to monitor how the search is converging toward the optimal-scoring tree.

2 Basics

2.1 Tree Fitness

We use the maximum parsimony (MP) criterion to compute the fitness of a phylogenetic tree. MP is an optimization problem for inferring the evolutionary history of different taxa, in which it is assumed that each of the taxa in the input is represented by a string over some alphabet. The symbols in the alphabet can represent nucleotides (in which case, the input are DNA or RNA sequences), or amino-acids (in which case the input are protein sequences), or may even include discrete characters for morphological properties. It is also assumed that the strings are put into a multiple alignment, so that they all have the same length. Maximum parsimony then seeks a tree, along with inferred ancestral sequences, so as to minimize the total number of evolutionary events (counting only point mutations).

Formally, given two sequences a and b of the same length, the *Hamming distance* between them is defined as $|\{i : a_i \neq b_i\}|$ and denoted as $H(a, b)$. Let T be a tree whose nodes are labeled by sequences of length k , and let $H(e)$ denote the Hamming distance of the sequences at each endpoint of edge e . The *parsimony length* of the tree T is $\sum_{e \in E(T)} H(e)$. The MP problem seeks the tree T with the minimum length; this is the same as seeking the tree with the smallest number of point mutations for the data. MP is an NP-hard problem [6], but the problem of assigning sequences to internal nodes of a fixed leaf-labelled tree is polynomial [5].

2.2 Robinson-Foulds Distance

In our experiments, we compare trees found by our Simple Local Search (SLS) algorithm to target trees for the data under consideration. We use the Robinson-Foulds (RF) distance to measure the topological distance between trees. The

RF distance between two trees is the number of bipartitions that differ between them. It is useful to represent evolutionary trees in terms of *bipartitions*. Removing an edge e from a tree separates the leaves on one side from the leaves on the other. The division of the leaves into two subsets is the bipartition B_i associated with edge e_i . Let $\Sigma(T)$ be the set of bipartitions defined by all edges in tree T . The RF distance between trees T_1 and T_2 is defined as

$$d_{RF}(T_1, T_2) = \frac{|\Sigma(T_1) - \Sigma(T_2)| + |\Sigma(T_2) - \Sigma(T_1)|}{2}$$

Our figures plot the *RF rate*, which is obtained by normalizing the RF distance by the number of internal edges and multiplying by 100. (Assuming n is the number of taxa, there are $n - 3$ internal edges in a binary tree). Hence, the RF rate varies between 0% and 100%. Two trees with no bipartitions in common will have an RF rate of 100%. Identical trees have an RF rate of 0%.

3 Fitness Distance Correlation

We compute the correlation between tree characteristics based on a measure proposed by Jones and Forrest for genetic algorithms [10]. Their measure computed the correlation between the fitness and Hamming distance between n individual solutions in a population. We extend their measure for use in a phylogenetic search. In particular, consider a set $F = \{f_1, f_2, \dots, f_n\}$ of fitnesses (or MP scores) and a corresponding set $D = \{d_1, d_2, \dots, d_n\}$ of n RF distances to a target tree. In our study, the target tree will either be the *best-tree-overall* or the *best-tree-so-far*.

We compute the correlation coefficient, r_{FD} , between the two sets F and D as

$$r_{FD} = \frac{c_{FD}}{\sigma_F \sigma_D}, \text{ where}$$

$$c_{FD} = \frac{1}{n} \sum_{i=1}^n (f_i - \bar{f})(d_i - \bar{d}) \quad (1)$$

is the covariance of F and D , and σ_F , σ_D , \bar{f} , and \bar{d} are the standard deviations and means of F and D , respectively.

Consider a set \mathcal{T} of phylogenetic trees. We would like to compute the r_{FD} correlation coefficient using the trees in \mathcal{T} . Each tree in \mathcal{T} will have a fitness value (or MP score). Those MP scores compose the set F . Moreover, the topology of each tree in \mathcal{T} will be compared to a target tree t . The topology of a pair of trees t_i and t_j is compared by computing their RF distance, $d_{RF}(t_i, t_j)$. Since we are interested in computing the RF distance between all trees in \mathcal{T} and a specific tree t , we perform a one-to-all comparison between the trees in \mathcal{T} and the target tree, t .

A strongly positive (or negative) r_{FD} coefficient, $-1 \leq r_{FD} \leq +1$, indicates that the solution quality gives good (bad) guidance searching for the target tree. r_{FD} values close to zero indicate no clear correlation between fitness and distance. The search is essentially wondering aimlessly in tree space.

4 Simple Local Search Heuristic

Our Simple Local Search (SLS) heuristic operates by successively exploring the neighborhood of a current solution and moving to one of its neighbors. First, SLS uses random sequence addition (RSA) to create the initial starting tree. To construct a RSA tree, we randomize the ordering of the sequences in the dataset. Afterwards, the first three taxa are used to create an unrooted binary tree, T . The fourth taxon is added to the internal edge of T that results in the best MP score. This process continues until all taxa have been added to the tree. Starting trees can also be based on neighbor-joining (NJ) [14] or by generating a starting tree randomly.

4.1 Tree Neighborhoods

Once we have a tree T , we improve it by rearranging its edges in a way that improves its maximum parsimony score. There are three main types of rearrangement operations, which defines the neighborhood of T , used in phylogenetic search heuristics [4].

- The *nearest-neighbor interchange (NNI)* operation swaps two adjacent branches on the tree. In other words, it erases an interior edge on the tree, and the two branches connected to it at each end (so that a total of five branches are erased). Afterwards, four subtrees are disconnected from each other. Four subtrees can be hooked together into a tree in three possible ways, where one of the trees is the original one. For a tree T with n taxa, $2(n-3)$ neighbors can be examined for each tree. Local searches based strictly on NNI operations perform poorly in comparison to their SPR and TBR counterparts.
- A *subtree pruning and regrafting (SPR)* move consists of removing an edge from the tree with a subtree attached to it. The subtree is then reinserted into the remaining tree in all possible places, each of which inserts a node into a branch of the remaining tree. Since there are n exterior edges and $n-3$ interior edges on an unrooted binary tree, the total number of solutions in the neighborhood is $4(n-3)(n-2)$.
- In a *tree-bisection and reconnection (TBR)* move, an interior branch is broken, and the two resulting fragments of the tree are considered as separate trees. All possible connections are made between a branch of one and a branch of the other. If there are n_1 and n_2 species in the subtrees, there will be $(2n_1-3)(2n_2-3)$ trees in a TBR neighborhood.

With a mechanism for generating a neighborhood, we must decide which neighboring tree T' should be selected. SLS uses a *first improvement algorithm* to select a neighbor. If $score(T') < score(T)$, then T' is accepted to replace the current tree T . The search continues until there is no neighbor T' with a better score than the current tree, T . If no better neighbor can be found, a local optimum has been reached and SLS terminates.

4.2 Search Path Trees

The search history of a phylogenetic heuristic is the set of neighbors selected along the search path to the best tree. Let β represent the type of move used in a neighborhood. Hence, $\beta \in \{NNI, SPR, TBR\}$. P_β denotes the search path trees consisting of selecting the first-improving neighbor from a β neighborhood. The sequence of trees encountered along the search path using a β operation is defined as

$$P_\beta = (t_1, \dots, t_m).$$

For a path P_β , the search examines tree t_i before tree t_j , where $0 \leq i < j \leq m$. There are m trees on the search path, P_β , where t_1 represents the initial (or starting) tree, and t_m is the final tree (e.g., local optimum). Thus, P_β represents the historical record of the phylogenetic search. All of the phylogenetic trees in P_β are binary trees.

5 Experimental Methodology

5.1 Molecular Sequences

We used the following biological datasets as input to study the behavior of our SLS heuristic.

1. A 44 taxa dataset (17,028 sites) of placental mammals that includes 19 nuclear and 3 mitochondrial gene sequences for 42 placental and 2 marsupial outgroups [11]. In our experiments, both SLS and PAUP* established a best score of 43,085.
2. A 60 taxa dataset (2,000 sites) of ensign wasps composed of three genes (28S ribosomal RNA (rRNA), 16S rRNA, and cytochrome oxidase I (COI)) [2]. Our SLS heuristic established a best score of 8,698 on this dataset.
3. A 174 taxa dataset (1,867 sites) of insects and their close relatives for the nuclear small subunit ribosomal RNA (SSU rRNA) gene (18S). The sequences were manually aligned according to the secondary structure of the molecule [7]. For this dataset, SLS established a best MP score of 7,440.

Since we do not know the true tree for these datasets, we use an approximation to the true tree. We run PAUP*, Phylip, and SLS on the above molecular sequences. The best-scoring tree(s) found by these heuristics is considered to be the “true tree”. Both SLS and PAUP found best-scoring trees with a maximum parsimony score of 43,085 on Dataset #1. Hence, those trees make up the *best-tree-overall* set for this dataset. For the remaining datasets, SLS found the best-scoring trees and as result establishes the *best-tree-overall* set for Datasets #2 and #3. Accuracy, as measured by the Robinson-Foulds (RF) distance is between the trees of interest and the *best-tree-overall* (or “true tree”). We also compute accuracy with the *best-tree-so-far* (described in more detail in 6.3) by the SLS heuristic.

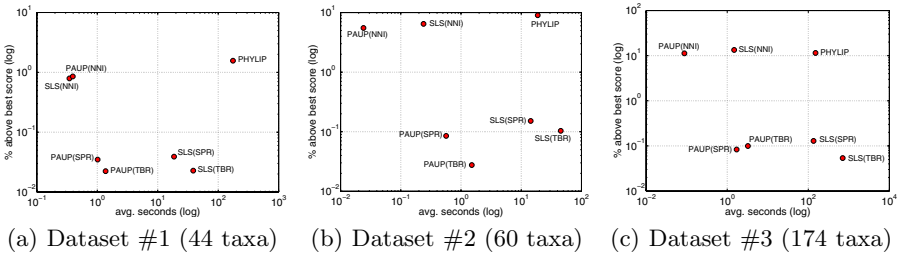


Fig. 1. Performance of SLS, PAUP*, and Phylip. The same set of random addition sequence starting trees was used by each heuristic. The best scores found for each of the datasets (from smallest to largest) is 43085, 8698, and 7440. Each data point represents the average of five runs.

5.2 Implementation and Platform

Our SLS algorithm is implemented in C++. Our implementation took advantage of the `libcov` [1] phylogenetic software package to handle reading data matrices plus we used their data structures to manipulate trees. However, we wrote our own branch-swapping routines as well as developed an algorithm for calculating the MP score more efficiently. We used the HashRF algorithm to compute the RF distances between trees [16]. Random trees were also generated using PAUP*. All algorithms were run five times on each of the biological datasets. All experiments were run on an Intel Pentium D platform with 3.0GHz dual-core processors and a total of 2GB of memory.

6 Experimental Results

6.1 SLS Performance

Figure 1 compares the performance of our SLS algorithm to local search heuristics implemented in PAUP* [17] and Phylip [3] on three biological datasets, which are described in detail in Section 5.1. The plots demonstrate that our SLS implementation performs comparably to PAUP* in terms of finding similar scoring MP trees. However, our algorithm does require more time to find good-scoring trees. Given that our objective of understanding the relationship between parsimony scores and topological distance, the actual runtime of a local search heuristic is not of concern here. Our objective is to make sure that the best-scoring trees found by SLS are comparable to those found by PAUP*. We note that Phylip is not competitive in terms of MP scores found to either PAUP* or SLS.

6.2 r_{FD} and the *Best-Tree-Overall*

Before exploring the behavior of the r_{FD} value, we take a look at the search path lengths of all the datasets and neighborhoods used in this study. Table 1 shows

Table 1. Total number of search path trees for the datasets under study

	$ P_{NNI} $	$ P_{SPR} $	$ P_{TBR} $
Dataset #1 (44 taxa)	166	224	265
Dataset #2 (60 taxa)	276	778	792
Dataset #3 (174 taxa)	748	1,698	1,870

Table 2. The fitness distance correlation coefficients (r_{FD}) for all three datasets

Fitness distance (r_{FD})				
	P_{NNI}	P_{SPR}	P_{TBR}	P_{RAND}
Dataset #1 (44 taxa)	0.84	0.81	0.89	0.41
Dataset #2 (60 taxa)	0.82	0.92	0.95	0.28
Dataset #3 (174 taxa)	0.94	0.96	0.98	0.21

the result. As the number of taxa in a dataset increases, the number of trees in the search path also increases. This corresponds to there being more trees in the search space to consider for larger datasets. Furthermore, larger neighborhoods (e.g., TBR) have more trees to consider than their smaller counterparts (e.g., NNI). The search path lengths are also important since they represent the number of values in the F and D sets used in Equation 1. For example, computing the r_{FD} value for the 265 search path trees based on an P_{TBR} neighborhood for Dataset #1, results in $|F| = |D| = 265$ values. F will contain the fitness (parsimony scores) of the 265 search path trees. D will represent the RF rate between each of the 265 search path trees to a target tree. In this subsection, the target tree will be the *best-tree-overall*.

Table 2 shows the r_{FD} correlation coefficient of the search path trees used in this study. Based on the table, there is a strong positive correlation between tree scores and their RF distance to the *best-tree-overall*. The same data that are used for computing the r_{FD} coefficient can be graphically displayed in the form of a fitness-distance plot. Figure 2 shows the results for each of the datasets using TBR-based search path trees (P_{TBR}). The plots clearly show that fitness and distance are linearly correlated. Hence, a phylogenetic search that only uses fitness as a guide does result in trees that are topologically close to the *best-tree-overall*.

On the other hand, trees randomly extracted from tree space have very low correlation when compared to the *best-tree-overall*. Figure 3 shows the scatterplot of the fitness-distance pairs based on 10,000 trees randomly selected from the search landscape. Here, the r_{FD} value decreases from 0.41 (44 taxa trees) to 0.21 (174 taxa trees). Based on this result, one would predict that the correlation between fitness and distance for random trees will tend toward zero as the taxa size increases. Hence, the r_{FD} value says that the search is wondering aimlessly in tree space, which matches exactly with the behavior of randomly selecting trees.

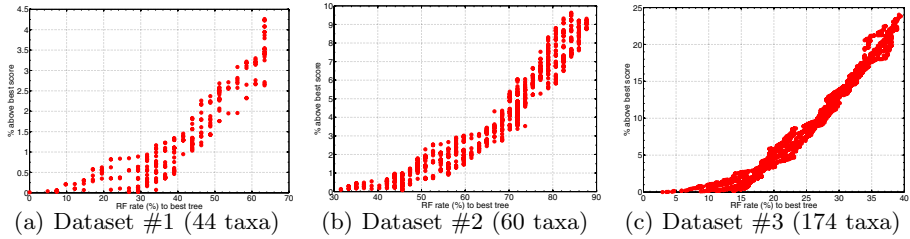


Fig. 2. Fitness distance correlation (r_{FD}) for P_{TBR} search path trees on each of the biological datasets. The fitness (MP score) and the RF rate is relative to the *best-tree-overall* tree of trees selected along the path to the local optimum under a TBR neighborhood.

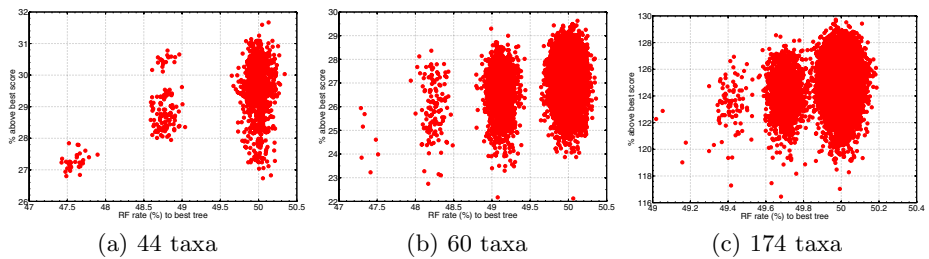


Fig. 3. Fitness distance correlation for 10,000 random trees, where the size of the trees is related to the number of taxa in the biological datasets. (a) $r_{FD} = 0.41$. (b) $r_{FD} = 0.28$. (c) $r_{FD} = 0.21$

6.3 r_{FD} and the *Best-Tree-So-Far*

One of the limitations of how the fitness-distance coefficient is used in other applications is that access to the best (or optimal) solution is required. In phylogenetics, for datasets that have been heavily studied such as the `rbcl500` (or Zilla data) [8], [12] this isn't necessarily a problem. Moreover, on the datasets used in this study, we have used numerous software packages (e.g., PAUP* and Phylip) to establish the *best-tree-overall*. However, it is likely that better trees do exist in tree space, which maybe be found as new phylogenetic heuristics are developed, for these datasets. However, suppose we don't have access to a reliable *best-tree-overall*? How can the r_{FD} correlation coefficient be of use in this situation?

As a phylogenetic heuristic progresses through the search landscape, it will always have access to the *best-tree-so-far*. In other words, if a search has been running for time α , the search can return the fitness of the best-scoring tree its seen for that particular time point. Our next experiment looks at the r_{FD} coefficient of the search at different time intervals (0%, 20%, ..., 100%) of the search. The 0% time interval (or search progress) represents the starting trees. By Equation 4.2, this represents tree t_0 in the search path. The 20% time interval

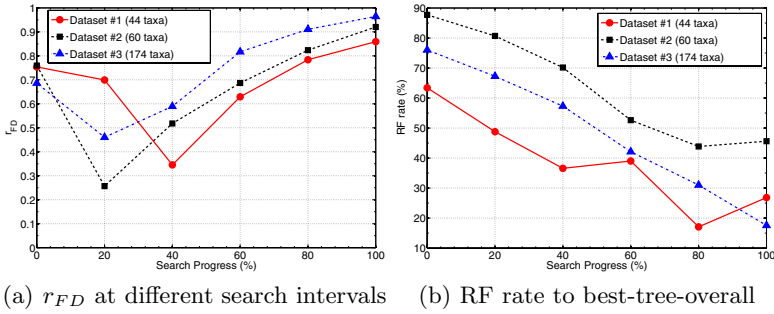


Fig. 4. r_{FD} are estimated with search path trees at each search progress on all dataset

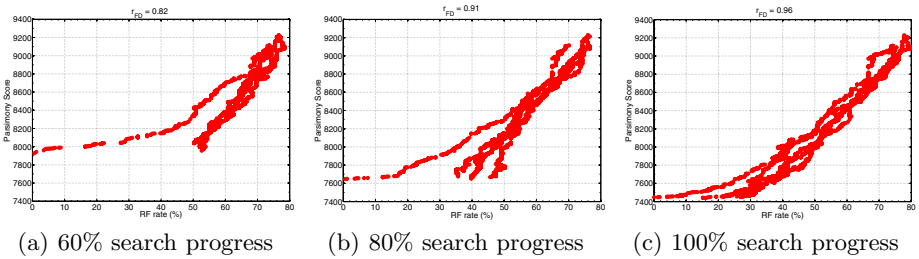


Fig. 5. r_{FD} at various points in the search

represents tree $0.20 \cdot m$, where m is the length of the search path. The remaining tree interval points are found similarly.

Figure 4 (a) shows the r_{FD} values based on different search intervals based on a TBR neighborhood. For example, to compute the r_{FD} values at 20% search progress, only trees labeled from t_0 to $t_{0.20 \cdot m}$ are used in the calculation. Furthermore, the RF distances between each of these trees is compared to the *best-tree-so-far*, that is the best-tree found within the 0% to 20% time interval.

r_{FD} values in Figure 4 (a) decrease in the beginning and increase again at the 40 % search mark. The initial r_{FD} values are high since there are not many points involved in the calculation. However, after 40 % search progress, the r_{FD} value steadily increases showing a high positive correlation. Figure 5 shows the scatter plots of the r_{FD} values for Dataset #3 at 60%, 80%, and 100% search progress. According to Figure 4(a), the r_{FD} values are strongly correlated by 80% search completion. If the search were to stop early, what would be effect on the topological accuracy of the search as it relates to the *best-tree-overall*. Figure 4(b) shows the results. For each point, the RF distance between the *best-tree-so-far* at $p\%$ search progress is compared with the *best-tree-overall*. Clearly, at 80% search progress for the two smallest datasets, there is minimal (if any) loss in topological accuracy when compared with completing the search (100% search progress). Furthermore, there is a savings of 20% in overall computational time.

7 Conclusions and Future Work

It is impossible to know the true evolutionary history for a set of organisms. As a result, phylogenetic heuristics attempt to find the optimal-scoring tree in the exponentially-sized tree space. Such a strategy is based on the assumption that better-scoring trees relates to depicting accurately the evolutionary relationships between the set of organisms.

We develop a new correlation coefficient called r_{FD} to quantify the relationship between fitness (MP scores) and distance (topological accuracy) of the trees found during a phylogenetic search. Based on a variety of different biological datasets, our results show that improvements in fitness are strongly correlated ($r_{FD} > 0.8$) with topological distance to the *best-tree-overall*. However, we also investigated the use of the r_{FD} coefficient if the best overall tree is not available. Every run of a phylogenetic search can produce a *best-tree-so-far*. By monitoring the search at different time intervals, we also found that the r_{FD} coefficient shows strong positive correlation. Hence, the r_{FD} value is robust in that it does need access to the *best-tree-overall*. As the search gets closer to terminating at a local optimum, the r_{FD} value increases accordingly. Hence, r_{FD} values could be used as stopping criterion to determine when a search should stop. For Datasets #1 and #2, it would be safe to stop early (at the 80% search progress point) without any penalties in topological accuracy. Furthermore, a savings of 20% in computation is saved without any corresponding loss in accuracy.

Future work will investigate the value of r_{FD} by examining additional datasets. The use of r_{FD} as a way to monitor convergence real-time during a search will also be studied. We plan to improve the performance (in terms of running time) of our SLS implementation and make it publicly available to the systematic community. Furthermore, we plan to apply our approach to more powerful heuristics such as parsimony ratchet [12], RAxML [15], and MrBayes [13] which will allow us to analyze much larger datasets.

Acknowledgments

This work was funded by the National Science Foundation under grant IIS-0713618. The authors wish to thank Bill Murphy and Matt Yoder for providing us with the biological datasets used in this study.

References

1. Butt, D., Roger, A., Blouin, C.: libcov: A C++ bioinformatic library to manipulate protein structures, sequence alignments and phylogeny. BMC Bioinformatics 6(138) (2005)
2. Deans, A.R., Gillespie, J.J., Yoder, M.J.: An evaluation of ensign wasp classification (Hymenoptera: Evanilidae) based on molecular data and insights from ribosomal rna secondary structure. Syst. Ento. 31, 517–528 (2006)
3. Felsenstein, J.: Phylogenetic inference package (PHYLIP), version 3.2. Cladistics 5, 164–166 (1989)

4. Felsenstein, J.: *Inferring Phylogenies*. Sinauer Associates (2003)
5. Fitch, W.M.: Toward defining the course of evolution: minimal change for a specific tree topology. *Syst. Zool.* 20, 406–416 (1971)
6. Foulds, L.R., Graham, R.L.: The Steiner problem in phylogeny is NP-complete. *Advances in Applied Mathematics* 3, 43–49 (1982)
7. Gillespie, J., McKenna, C., Yoder, M., Gutell, R., Johnston, J., Kathirithamby, J., Cognato, A.: Assessing the odd secondary structural properties of nuclear small subunit ribosomal rna sequences (18s) of the twisted-wing parasites (Insecta: Strepsiptera). *Insect Mol. Biol.* 15, 625–643 (2005)
8. Goloboff, P.: Analyzing large data sets in reasonable times: solutions for composite optima. *Cladistics* 15, 415–428 (1999)
9. Goloboff, P.A., Farris, J.S., Nixon, K.C.: TNT, a free program for phylogenetic analysis. *Cladistics* 24(5), 774–786 (2008)
10. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: Eshelman, L. (ed.) *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp. 184–192. Morgan Kaufmann, San Francisco (1995)
11. Murphy, W.J., Eizirik, E., O'Brien, S.J., Madsen, O., Scally, M., Douady, C.J., Teeling, E., Ryder, O.A., Stanhope, M.J., de Jong, W.W., Springer, M.S.: Resolution of the early placental mammal radiation using bayesian phylogenetics. *Science* 294, 2348–2351 (2001)
12. Nixon, K.C.: The parsimony ratchet, a new method for rapid parsimony analysis. *Cladistics* 15, 407–414 (1999)
13. Ronquist, F., Huelsenbeck, J.P.: Mrbayes 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics* 19(12), 1572–1574 (2003)
14. Saitou, N., Nei, M.: The neighbor-joining method: A new method for reconstruction phylogenetic trees. *Mol. Biol. Evol.* 4, 406–425 (1987)
15. Stamatakis, A., Ludwig, T., Meier, H.: RAxML: A fast program for maximum likelihood-based inference of large phylogenetic trees. *Bioinformatics* 1(1), 1–8 (2004)
16. Sul, S.-J., Williams, T.L.: An experimental analysis of robinson-foulds distance matrix algorithms. In: Halperin, D., Mehlhorn, K. (eds.) *ESA 2008. LNCS*, vol. 5193, pp. 793–804. Springer, Heidelberg (2008)
17. Swofford, D.L.: *PAUP**: Phylogenetic analysis using parsimony (and other methods), Sinauer Associates, Underland, Massachusetts, Version 4.0 (2002)

Alignment and Analysis of Closely Related Genomes

Allison Regier, Michael Olson, and Scott J. Emrich*

Computer Science and Engineering, University of Notre Dame
Notre Dame, IN 46556
{aregier,molson3,semrich}@nd.edu

Abstract. Complex strategies have been developed for whole genome alignment of multiple species. In the case of population genomics studies, the sequences being aligned are often very similar. We propose a reference coordinate system that simplifies the task of comparing many closely related genomes while taking into account structural rearrangements. We implemented software to compare a group of three strains of the malaria mosquito, *Anopheles gambiae* and a group of three strains of the malaria parasite, *Plasmodium falciparum*. Our simplified representation enables us to leverage existing work on the species while performing fine-grained analysis on the new draft genomes. Further, this approach will easily scale to hundreds of closely related genomes, enabling new analyses in population genomics as additional genomic sequences become available.

1 Introduction

Now that genome sequencing has become less expensive, biologists are able to ask even broader questions about genetic variation. Computational approaches for studying variation between two genome sequences often utilize whole genome alignment (WGA) [1]. Because of the historical diversity of completed genomes, genome alignment methods have been designed to be sensitive enough to find homologies between diverged genomes such as human and mouse [2]. Most run on a typical workstation using an assortment of fast alignment approximation techniques.

Given the extent of genomic rearrangement in diverged genomes, complex strategies have been developed to extend whole genome alignment to multiple genomes. Most WGA approaches overcome rearrangements by constructing a “homology map”, which determines collinear blocks between two or more genomes. Alignment is then restricted to these collinear blocks. In addition, regions defined in a homology map can be processed using traditional multiple alignment tools, which allows WGA methods to use previously developed programs to perform the comparison [3,4,5,6]. These methods have been useful in gene finding and other applications [7,8,9,10].

* Corresponding author.

Here, we consider a special case of WGA in the context of population genomic studies where draft quality sequencing is performed on related individuals, subspecies, or clinical isolates. Specifically, this work is motivated by ongoing efforts in malaria biology where tens of mosquito genomes and hundreds of malaria parasite genome projects are planned to help eradicate this deadly disease. Because each draft genome may be segmented in hundreds or thousands of pieces, these regions need to be consistently arranged prior to analysis. Further, in the case of malaria-focused efforts, at least one closely related genome has been highly finished and is available in chromosome form. In these cases it is desirable to leverage existing structural and annotation knowledge about this genome that will generally hold true in related genomes. Subsequent analysis is simplified because the need to annotate each genome separately is eliminated. Annotated gene structures can be easily correlated with rates of genetic variation to provide important information about evolutionary processes.

In this work, we develop an alternative representation for WGA that takes advantage of the observation that closely related genomes contain only limited rearrangement. Rather than describing an alignment in terms of a dynamic homology map, we choose a specific reference coordinate system based on a highly finished genome. This simple approach allows one or more sets of scaffolds from closely related genomes (perhaps hundreds!) to be compared in a straightforward manner for population genomics studies. We develop a module for a popular WGA tool that generates an estimated chromosome-anchored draft genome based on the reference coordinate system. This facilitates comparing tens of genomes directly using the vertical multiple alignment (VMA) format [11]. We apply our pipeline to two groups of organisms: a group of three strains of the malaria mosquito (*Anopheles gambiae*) and a group of three strains of the malaria parasite (*Plasmodium falciparum*). As additional genomes from these species become available, our approach will scale to large-scale population genomics projects.

2 Related Work

Previously, genomes were compared using experimental techniques. For example, fluorescent in situ hybridization (FISH) can be used to roughly anchor draft scaffolds on a chromosome even if no prior information is available about the genome being sequenced. Genetic maps are another form of experimental data that can be used to anchor scaffolds onto chromosomes in a specific order and orientation. However, both of these techniques require expensive, manual intervention. If a closely related genome is available, then computational WGA is a more practical technique that requires no additional experimentation.

WGA is an extension of the classic bioinformatics problem of sequence alignment. In general, the goal of sequence alignment is to find similar regions between sequences. Usually sequence alignment is classified as either global [12] or local [13]. Finding an optimal alignment is infeasible for long sequences, so heuristic algorithms have been developed [14,15,16,17,18,19,20]. The availability of whole genome sequences introduces a number of new challenges. Not only do algorithms

need to be efficient to align whole genome sequences, but they also must take both nucleotide mutations and structural rearrangements into account. Some methods assume that orthologous regions are known in advance, so these don't need to consider structural rearrangements [21]. Other whole genome aligners have been designed that use aspects of both global and local alignment to better represent the relationship between two whole genome sequences [2,22,23,24,25]. First, local alignments are used to find seeds. Next, seeds that are consistent with each other are chained or clustered together to form the alignments. Whole genome alignment has also been extended to align multiple genomes of various degrees of divergence [3,4,5,6,8].

Most tools provide alignment summaries consisting of chromosome or scaffold names, start and end positions, relative orientations of the sequences in the alignment, and measures of similarity such as percent identity. This information can be displayed in a browser [26,27] where one sequence is chosen as the reference sequence and the alignment is shown relative to that reference coordinate system. This can be a useful interface for scientists who want to visualize a specific region of the genome, for example one containing a gene of interest. However, if further computational processing is needed, for example an analysis of variation across the chromosome, the alignment format can be quite complex to deal with. When many closely related genomes are being compared, as in a population genetics study, a simpler representation is needed.

The work here was in part inspired by the syntenic assembly used in a recent population genomics study of *Drosophila simulans* [10]. In this case, the reference sequence chosen was the finished genome of *D. melanogaster*. Reads from several strains of *D. simulans* were aligned to the reference genome, then the alignment was iteratively refined. Additionally, draft scaffolds of the outgroup for the study, *D. yakuba*, were aligned to the reference sequence using Mercator [6] and MAVID [20]. The results of the alignment were represented in VMA format. In the organisms of interest to our malaria efforts, draft scaffolds of the query genomes had already been assembled. Therefore, we independently developed a pipeline that uses genome alignment information to construct an estimated chromosome-anchored draft genome for our target organisms. The estimated genome uses the content from a query genome, but this content is translated into the coordinate system of a reference genome. The estimated chromosomes can be represented simply as a FASTA file, making it easy to manipulate for further analysis. Alternatively, estimated chromosomes from many query genomes can be represented in VMA format, which allows simultaneous comparisons between multiple closely related genomes.

3 Pipeline

Our pipeline consists of three main steps: alignment, filtering, and translation. A schematic is presented in Fig. 1. First, each sample genome is aligned to the reference genome using a whole genome alignment technique. There are several existing software packages that can be used for whole genome alignment. Any

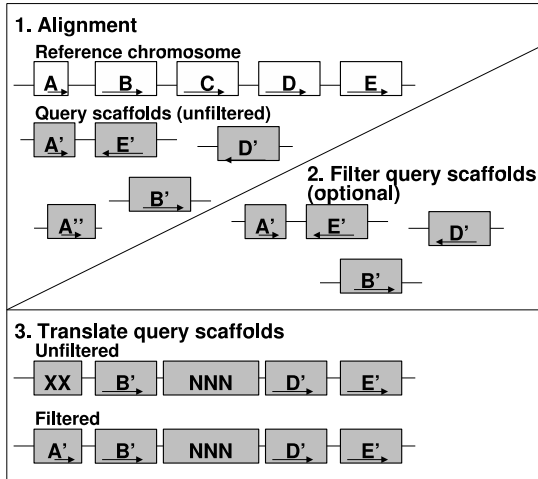


Fig. 1. Pipeline to translate the content of query scaffolds into coordinates of a reference chromosome

whole genome alignment algorithm can be used in the first step of our pipeline. The alignment software identifies homologous blocks between a pair of genomes.

Next, smaller scaffolds are optionally filtered out. Filtering may need to be adjusted based on the quality of the scaffolds and the degree of relatedness between genomes. For example, an assembly may contain scaffolds that are simply alternative haplotypes for a single portion of the genome. Ideally, each position in a scaffold would have a one-to-one mapping to a position on the reference chromosome. If alternative haplotypes have been assembled separately, then two scaffolds would align to a single location on the reference chromosome. The filtering process can choose which of these scaffolds should be used for further analysis. We should also keep track of the alternative haplotype if this information is needed later.

Finally, using the filtered alignment, each position in the set of scaffolds is translated into a corresponding position in reference coordinates. Reference coordinates with no corresponding query content are filled with the placeholder 'N'. Reference coordinates with two or more corresponding query sequences are filled with the placeholder 'X'. The result is a set of chromosomes for each sample genome. The chromosomes can be compared to each other base by base.

4 Methods

We implemented the pipeline using the whole genome alignment tool *nucmer* to provide the foundation for anchoring query genomes to a fixed reference. We created a module to filter and format the output from *nucmer* into estimated chromosomes so that the query genomes could easily be compared and analyzed.

```

Reference: A C T G A C T - G A C T G
Query:    A C - G C C T T G A C A G
Delta file:
>r_seqID q_seqID 100000 2000
40120 40132 1010 1022 4 2 0
3
-5
0

```

Fig. 2. Nucmer uses the delta file format to represent gapped alignments between homologous regions. The alignment can be reconstructed based on the gap information provided in the file.

Alignment. Nucmer is an alignment tool that was designed to be efficient enough to align long genomic sequences such as whole genomes, and flexible enough to detect homologous sequences at various degrees of divergence [25]. Nucmer performs a pairwise alignment between one or more query sequences (i.e., scaffolds in a draft genome) and one or more reference sequences (i.e., chromosomes in a finished genome). First, it anchors the alignment by finding maximal matches between the two genomes. Next, it clusters these matches together. Finally, it extends the matches into alignments using dynamic programming. Nucmer represents the alignments in a delta file, which encodes the locations of gaps in the alignment relative to the previous gap [28]. The delta file format is summarized in Fig. 2.

The nucmer delta file counts how many positions in the non-gapped sequence there are between the previous gap and the current one. Gaps appearing in the query sequence are labeled with a positive number, and gaps appearing in the reference sequence are labeled with a negative number. For example, the optimal alignment between the two sequences in the figure contains two gaps. The first two lines are headers to identify the sequences, the start and end positions of the alignments, and mismatch information. The following three lines contain gap information. The first gap occurs 3 positions from the beginning of the alignment in the query sequence, and the second gap occurs 5 positions later in the reference sequence.

The nucmer pipeline has many options to tailor the alignment to specific needs. For the *A. gambiae* analysis, we used the `-mum` option, which limits alignment seeds to only those that are unique in both genome sequences, and we set the minimum cluster size for an alignment to be 200 nucleotides. This cut down on spurious alignments that otherwise occur due to the repetitive nature of the genomes. For the *P. falciparum* analysis, we used the default nucmer settings: anchors must be unique in the reference sequence but not necessarily the query, and minimum cluster size was 65 nucleotides.

Filtering. We implemented an optional filtering step that discards scaffolds less than 100,000 nucleotides long. We used this filtering to reduce the number of reference positions that mapped to multiple query sequences, thus reducing ambiguity in the mapping.

Alternative filtering methods are possible and may be superior. One possibility is to translate all query scaffolds and, in case of ambiguity, to dynamically select a single scaffold for any given reference position based on some criterion. For example, if two scaffolds overlap on the same reference positions, it may be desirable to use the longer scaffold in the estimated chromosome and classify the shorter scaffold as a potential alternative haplotype.

Translation. We created a module that uses the delta file to translate query coordinates into reference coordinates. First, each alignment is loaded from the delta file. The alignments are sorted in two ways: by reference sequence and by query sequence.

In order to map a single coordinate from the query to the reference, the set of delta alignments for the given query sequence is retrieved, and a binary search is performed to find the delta alignment that contains that coordinate. We step through that alignment from the beginning, keeping track of the alignment gaps as they occur. Once the desired query coordinate is reached, the corresponding reference coordinate is output. If the coordinate is not found in any alignment, -1 is returned.

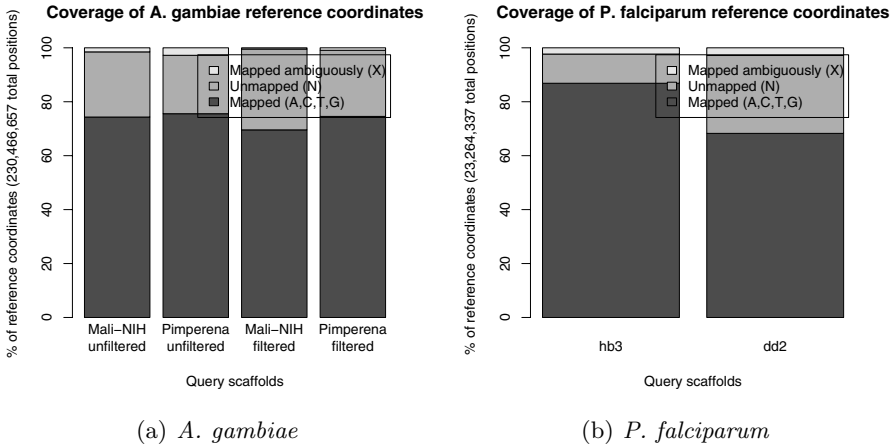
To map a set of coordinates in the same query sequence, the search coordinates are first sorted. Then, the above method is used to map the first coordinate in the set. When a coordinate is found, its reference location is printed, and the search coordinate is updated to be the next one in the set. We continue walking through the alignment from the same place, looking for the new search coordinate. If not all coordinates have been mapped when the end of a delta alignment is reached, the next delta alignment for this sequence is retrieved and the search continues.

Once all of the coordinates in the query genome have been mapped to coordinates in the reference genome, the translation module is used to create a new chromosome representation for the query genome. First, an empty chromosome is initialized with a position for each coordinate on the reference chromosome. Initially, all positions contain the placeholder 'N'. As each query scaffold position is translated, any information associated with that position (e.g., base call, quality information, SNP calls, etc.) are filled in to the reference coordinates. If multiple query coordinates map onto a single reference coordinate, either one of the coordinates is dynamically selected (see Sect. 4), or the ambiguity is indicated by placing an 'X' in that position.

The results can be output in two ways. First, each query genome can be printed as a multi-fasta file, with one sequence per reference chromosome. Each estimated chromosome in the multi-fasta file contains exactly the same number of nucleotides as its corresponding chromosome in the reference genome. Alternatively, one or more query genomes can be printed in a VMA format (see Table 1).

Table 1. VMA excerpt from *P. falciparum* Chromosome 5 alignment

Position	3D7	hb3	dd2
37022	C	T	C
37034	T	C	C
37035	A	T	T
37045	A	C	C
37048	T	G	G
37062	T	G	G
37068	A	G	G
37070	C	T	T
37073	T	A	A
37099	T	C	C
37100	C	T	T

**Fig. 3.** a) Results from *A. gambiae* estimated chromosomes. b) Results from *P. falciparum* estimated chromosomes. Proportion of bases mapped, unmapped, or mapped ambiguously.

5 Results

We used this pipeline to construct alignments for two different groups of genomes. The *A. gambiae* group consisted of a finished assembly of the PEST strain as well as two draft genomes from strains called Mali-NIH and Pimperena. There are 230,466,657 total bases in the 5 reliably assembled chromosome arms of the PEST assembly. Over 70% of the PEST reference coordinates are covered by uniquely mapped query sequence from both strains. When scaffolds shorter than 100,000 nucleotides are discarded, the number of reference positions with ambiguously mapped content (represented by the placeholder X) decreased slightly, but the number of reference positions with no content (represented by the placeholder N) increased. The number of unambiguously mapped reference

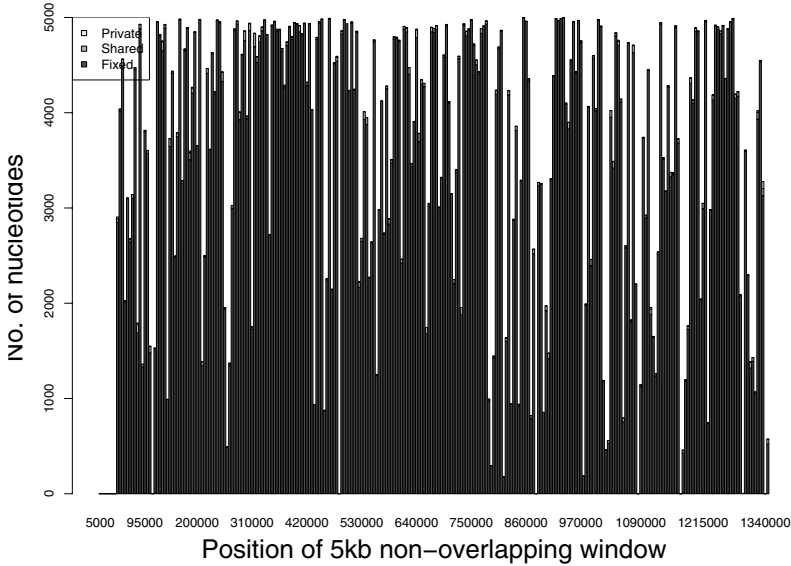


Fig. 4. Diversity across Chromosome 5 in *P. falciparum* based on VMA. Most nucleotides were fixed, i.e., 3D7 = hb3 = dd2. For positions where there is a single nucleotide polymorphism (SNP), these are broken down into shared differences where either hb3 or dd2 are the same as 3D7 or private differences, where 3D7 is not equal to either draft.

coordinates (A,C,T,G) is highest when no scaffold filtering is used. The coverage results are summarized in Fig. 3(a)

The *P. falciparum* group contained a finished reference genome from a strain of the malaria parasite known as 3D7 as well as draft genomes from two additional strains known as hb3 and dd2. There are 23,264,337 total bases in the 14 assembled chromosomes of the 3D7 strains. The filtering step was omitted, as the assembly was not expected to contain alternative haplotypes. The hb3 strain uniquely covers 87% of the reference coordinates, while the dd2 strain covers 68%. This difference in coverage levels could be explained by a number of factors. The assembly quality of the hb3 scaffolds could be higher, or the hb3 and 3D7 strains could be more closely related, and thus easier to align, than the dd2 and 3D7 strains. The coverage results are summarized in Fig. 3(b). A sample analysis of sequence diversity across chromosome 5 is shown in Fig. 4, which indicates most nucleotides are the same between all malaria isolates.

6 Conclusion

In this work, we propose and implement a simplified representation for WGA when the genomes being aligned are closely related, as in a population genomics study. Rather than maintaining a dynamic homology map, we choose a single reference coordinate system and translate all query sequences into that

coordinate system. In this way, existing work on a finished genome can be leveraged, and comparative analysis of multiple genomes at a nucleotide level becomes trivially scalable. Although the output will not show any structural variation, this information can be noted during the process of creating estimated chromosomes.

If the genomes are very similar on a nucleotide level, as is found in malaria isolates and mosquito subspecies, pairwise alignments with a single reference genome as outlined in this work is sufficient for population genomic analysis. If genomes are more diverged, we note that the estimated chromosome creation step could be done using more sophisticated progressive multiple alignment, even in the presence of limited genome rearrangements. This is a direction for future work.

Software is available on request.

Acknowledgments. This work was supported in part by a Lilly fellowship. We would like to thank our collaborators at the University of Notre Dame, the J. Craig Venter Institute, UC Davis, and Imperial College London.

References

1. Feuk, L., et al.: Discovery of human inversion polymorphisms by comparative analysis of human and chimpanzee DNA sequence assemblies. *PLoS Genet.* 1(4), 56 (2005)
2. Kent, W.J., et al.: Evolution's cauldron: Duplication, deletion, and rearrangement in the mouse and human genomes. *Proceedings of the National Academy of Sciences* 100(20), 11484–11489 (2003)
3. Darling, A.C., et al.: Mauve: Multiple alignment of conserved genomic sequence with rearrangements. *Genome Res.* 14(7), 1394–1403 (2004)
4. Ovcharenko, I., et al.: Mulan: Multiple-sequence local alignment and visualization for studying function and evolution. *Genome Res.* (2004) gr.3007205
5. Blanchette, M., et al.: Aligning Multiple Genomic Sequences With the Threaded Blockset Aligner. *Genome Res.* 14(4), 708–715 (2004)
6. Dewey, C.: Whole-genome alignments and polytopes for comparative genomics. PhD Thesis (2006)
7. Kellis, M., et al.: Sequencing and comparison of yeast species to identify genes and regulatory elements. *Nature* 423(6937), 241–254 (2003)
8. Ma, J., et al.: Reconstructing contiguous regions of an ancestral genome. *Genome Res.* 16(12), 1557–1565 (2006)
9. Stark, A., et al.: Discovery of functional elements in 12 *Drosophila* genomes using evolutionary signatures. *Nature* 450(7167), 219–232 (2007)
10. Begun, D.J., et al.: Population genomics: Whole-genome analysis of polymorphism and divergence in *Drosophila simulans*. *PLoS Biology* 5(11), e310 (2007)
11. *D. simulans* syntenic assembly, http://www.dpgp.org/syntenic_assembly/
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. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. *Journal of Molecular Biology* 147, 195–197 (1981)

14. Schwartz, S., et al.: PipMaker—A Web Server for Aligning Two Genomic DNA Sequences. *Genome Res.* 10(4), 577–586 (2000)
15. Schwartz, S., et al.: Human-mouse alignments with BLASTZ. *Genome Res.* 13(1), 103–107 (2003)
16. Brudno, M., et al.: Fast and sensitive multiple alignment of large genomic sequences. *BMC Bioinformatics* 4, 66 (2003)
17. Delcher, A.L., et al.: Fast algorithms for large-scale genome alignment and comparison. *Nucl. Acids Res.* 30(11), 2478–2483 (2002)
18. Haas, B.J., et al.: DAGChainer: A tool for mining segmental genome duplications and synteny. *Bioinformatics* 20(18), 3643–3646 (2004)
19. Bray, N., et al.: AVID: A Global Alignment Program. *Genome Res.* 13(1), 97–102 (2003)
20. Bray, N., Pachter, L.: MAVID: Constrained Ancestral Alignment of Multiple Sequences. *Genome Res.* 14(4), 693–699 (2004)
21. Brudno, M., et al.: LAGAN and Multi-LAGAN: Efficient Tools for Large-Scale Multiple Alignment of Genomic DNA. *Genome Res.* 13(4), 721–731 (2003)
22. Brudno, M., et al.: Glocal alignment: Finding rearrangements during alignment. *Bioinformatics* 19(suppl. 1), i54–i62 (2003)
23. Pevzner, P., Tesler, G.: Genome rearrangements in mammalian evolution: Lessons from human and mouse genomes. *Genome Res.* 13(1), 37–45 (2003)
24. Couronne, O., et al.: Strategies and Tools for Whole-Genome Alignments. *Genome Res.* 13(1), 73–80 (2003)
25. Kurtz, S., et al.: Versatile and open software for comparing large genomes. *Genome Biol.* 5(2) (2004)
26. Frazer, K.A., et al.: VISTA: Computational tools for comparative genomics. *Nucl. Acids Res.* 32(suppl. 2), W273–W279 (2004)
27. Miller, W., et al.: 28-Way vertebrate alignment and conservation track in the UCSC Genome Browser. *Genome Res.* (2007) gr.6761107
28. MUMmer 3 manual, <http://mummer.sourceforge.net/manual/>

Computational Prediction of Genes Translationally Regulated by Cytoplasmic Polyadenylation Elements

Eric C. Rouchka¹, Xiangping Wang², James H. Graham³, and Nigel G.F. Cooper²

¹ Department of Computer Engineering and Computer Science, J.B. Speed School of Engineering, 123 JB Speed Building, University of Louisville, Louisville, Kentucky USA
eric.rouchka@louisville.edu

² Department of Anatomical Sciences and Neurobiology, 500 South Preston Street, University of Louisville, Louisville, Kentucky USA
x0wang04@louisville.edu, nigelcooper@louisville.edu

³ Department of Electrical and Computer Engineering, J.B. Speed School of Engineering, University of Louisville, Louisville, Kentucky USA
jhgrah01@louisville.edu

Abstract. Cytoplasmic post-transcriptional modification of mRNA transcripts in the form of polyadenylated (poly(A)) tails plays a key role in their translational control. The timing and degree of polyadenylation has been shown to be due in part to a consensus nucleotide sequence -- cytoplasmic polyadenylation elements (CPEs) which can be detected by a polyadenylation element binding protein (CPEB). An individual mRNA transcript controlled by CPEB may contain one or more CPE sites occurring upstream of a consensus hexamer poly(A) signal. A probabilistic model, CPEDetector, is presented for predicting whether or not a gene's translation is mediated by CPEB. CPEDetector takes into account detected CPE sites, poly-A sites, and distance metrics between the detected locations. This approach is tested against the 3' untranslated regions (UTRs) of known genes using the UTRdb database.

Keywords: CPE, CPEB, bioinformatics, hidden Markov model, context free grammar, untranslated region.

1 Introduction

1.1 Central Dogma and Gene Regulation

The Central Dogma of Molecular Biology in summary states that the process of creating a protein encoded by a gene first begins with the genomic DNA, which is transcribed into an RNA intermediary template, known as messenger RNA (mRNA), which in turn is translated into a protein sequence using the genetic code. Regulation of genes can occur at either the transcriptional level, through complexes that form at transcription factor binding sites, or at the translational level. Transcription factor binding sites are typically found upstream (5') of the transcription start site (TSS). In contrast, many translational control regulators bind to the 3' untranslated region (UTR) downstream of the coding region. One such translational control mechanism is cytoplasmic polyadenylation.

1.2 Cytoplasmic Polyadenylation and Translational Control

In a number of molecular processes such as oogenesis, embryogenesis, and synaptic plasticity of the central nervous system, it is important to have the mRNA available temporally and spatially for specific and efficient protein production. One important mechanism underlying such translational control is cytoplasmic polyadenylation.

It has been long known that all eukaryotic pre-mRNAs undergo polyadenylation in the nucleus before they are exported to the cytoplasm. The newly synthesized pre-mRNA is endonucleolytically cleaved at about 10 nucleotides upstream of the polyadenylation signal (PAS), a hexamer sequence found in the 3' UTR. The poly(A) stretch (typically around 30nt in length) is then added to the newly formed 3' end. The poly(A) tail and its bound proteins are important for termination of transcription, export of the mRNA from the nucleus, and protection of the mRNA from degradation by exonuclease. Multiple variants of the PAS sequence have been identified in nature with different occurrences and distinct efficiencies for polyadenylation (table 1) [1].

Table 1. Alternative hexamer polyadenylation site (PAS) patterns

Hexamer	Frequency	Hexamer	Frequency
AAUAAA	0.6431	UUUAAA	0.0133
AUUAAA	0.1645	AAGAAA	0.0122
UAUAAA	0.0354	AAAAAG	0.0088
AGUAAA	0.0298	AAUGAA	0.0088
AAUAUA	0.0188	AAUAGA	0.0077
GAUAAA	0.0144	ACUAAA	0.0066
CAUAAA	0.0144	AAAACA	0.0055
AAUACA	0.0133	GGGGCU	0.0033

The short poly(A) tails added during nuclear polyadenylation helps to stabilize the mRNA and prevent it from degradation. However, the elongation of poly(A) tails in the cytoplasm (cytoplasmic polyadenylation) has a rather different role: to recruit the mRNA for translation. Cytoplasmic polyadenylation was first identified in eggs and single-cell embryos [2–5], where little RNA transcription activity was detected. Polyadenylation was immediately followed by translation. Timed expression of pre-stored mRNA has also been observed during early embryonic development to initiate mitosis and in some circumstances, to dictate the polarity of the embryo. Such translational activation is accompanied by elongation of the poly(A) tails [6,7]. The

Table 2. Alternative cytoplasmic polyadenylation element site (CPE) patterns

CPE Pattern
UUUUUAU
UUUUUGU
UUUUUACU
UUUUUGUU
UUUUAAU
UUUUACU
UUUUAAU

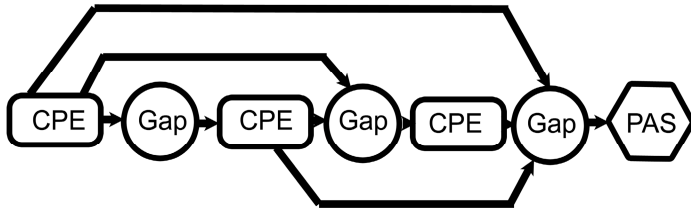


Fig. 1. Description of elements in 3' UTRs involved in translational regulation. Rounded rectangles, circles, along with the hexagon represent the various states. Each state is in turn a probabilistic model. Lines represent possible transitions from each probabilistic state. CPE: cytoplasmic polyadenylation element. Gap: nucleotide sequence gap. HEX: 6-base poly(A) signal.

same mechanism was later discovered to be employed by the hippocampus and several other parts of the central nervous system, for the acquisition and consolidation of memory, or for long-term synaptic plasticity.

Both a PAS motif and a U-rich motif are required for cytoplasmic polyadenylation of local mRNAs [8,9]. This short U-rich sequence UUUUUAU is known as the “cytoplasmic polyadenylation element” (CPE) [8]. Several sequence variants of CPE have been identified [8,10] (Table 2). CPE is often located upstream of the PAS, although the distance to PAS usually varies between ~10nt and 100nt, and in some instances CPE overlaps with the PAS. Many mRNAs have more than one CPE in their 3' UTR. The sequence, the number of copies of CPE and the distance to the PAS are widely variable for different mRNAs. Such variables could indicate the presence of modulators of the process of translational control. Fig. 1 illustrates the different signals found in mRNAs regulated by cytoplasmic polyadenylation element binding proteins.

1.3 Cytoplasmic Polyadenylation Element

Cytoplasmic polyadenylation element binding protein (CPEB) was first identified in 1990 in *Xenopus* oocytes [10,11]. The term CPEB became interchangeable with CPEB1 later as more paralogs were identified [12–16]. Studies on the mechanisms of CPEB-regulated translational control have extensively focused on CPEB1. Although evidence suggests that similar machineries are employed in oocytes, early embryos, and CNS, the major discoveries were derived from work in oocytes, since oocyte maturation is a more facile and efficient model for CPEB1-regulated translational control.

CPEB1 has a dual role as a translational activator and a translational inhibitor. Under quiescent conditions, unphosphorylated CPEB1 binds to the 3' UTR of mRNA and represses translation. When cells respond to appropriate stimuli, the activation of CPEB1 via phosphorylation or the degradation of CPEB1 removes the repression and allows translation to initiate. The complete mechanism through which CPEB1 regulates polyadenylation is yet to be completely elucidated, but current knowledge indicates that it involves both the 3' and 5' end of the mRNAs. In the dormant state, an adaptor protein, maskin, interacts with both CPEB1 (which binds to CPE in the 3'

UTR of the mRNA) and translation initiation factor eIF4E (which binds to 5' cap of the mRNA) [17]. This binding bends the mRNA molecule to bring its 3' and 5' ends close together. The interaction of maskin to eIF4E excludes eIF4G from binding to eIF4E. This blocks the assembly of the complete translational machinery. Phosphorylated CPEB is transformed into an activator which gains enhanced affinity to Cleavage and Polyadenylation Specific Factor (CPSF), which in turn recruits Poly (A) Polymerase (PAP) and initiates polyadenylation. The newly synthesized poly (A) tail is quickly bound and protected by Poly (A) Binding Protein (PABP), which subsequently recruits translational initiation factor eIF4G to displace maskin from eIF4E. This stabilized eIF4E-4G interaction enrolls the ribosome for initiation of translation.

CPEB regulated translation of CPE-containing mRNAs is a temporally and/or spatially tightly controlled process. It is the common underlying foundation for oogenesis, embryogenesis and synaptic plasticity. Both CPEB protein and its target mRNAs need to be localized to the proper location for the local control of translation. The cis-element CPE in the 3' UTR of mRNA molecules and the RNA-binding protein CPEB have been demonstrated in the transport of such mRNAs. A review of the role of CPEB is given in [18–20].

2 Methods

Probabilistic models have been applied to classify a number of potentially meaningful biological signals, particularly those involved in regulation. Examples include Gibbs sampling [21–23] and expectation-maximization approaches [24] for locating sequence motifs; hidden Markov models (HMMs) for detecting motifs and protein family classifications [25,26] as well as gene prediction [27,28]; and stochastic context free grammars (SCFGs) for detecting RNA genes [29,30]. While much remains to be uncovered about genes translationally regulated by CPEB, patterns regarding the RNA-recognition motif are beginning to emerge [31]. Using this information as a first step, a stochastic context-free grammar is created to identifying potential targets of CPEB.

Our model, CPEDetector, creates a finite state automata (FSA) based on the components shown in Fig. 1, which includes cytoplasmic polyadenylation elements (CPE), a poly(A) signal (PAS), and nucleotide gaps (GAP) occurring between elements. Each of these components is an independent probabilistic model (described in detail in sections 2.1-2.3), resulting in a single emission probability for each component in the FSA. Transition probabilities between states in this case are treated as one. Initially, each input sequence (a fasta sequence that is typically a 3' UTR region of a gene of interest) is scanned and scored, nucleotide by nucleotide, for the presence of CPE and PAS. Once the independent components resulting in a probability score greater than zero are found, they are put together in a model.

2.1 CPE Modeling

Cytoplasmic polyadenylation elements are modeled according to the observed functional CPE sites [8,10]. The consensus model sequence is UUUUUGU. However, alternative, functional CPE sites containing either seven or eight nucleotides have been found. Fig. 2 shows the FSA for modeling CPE sites.

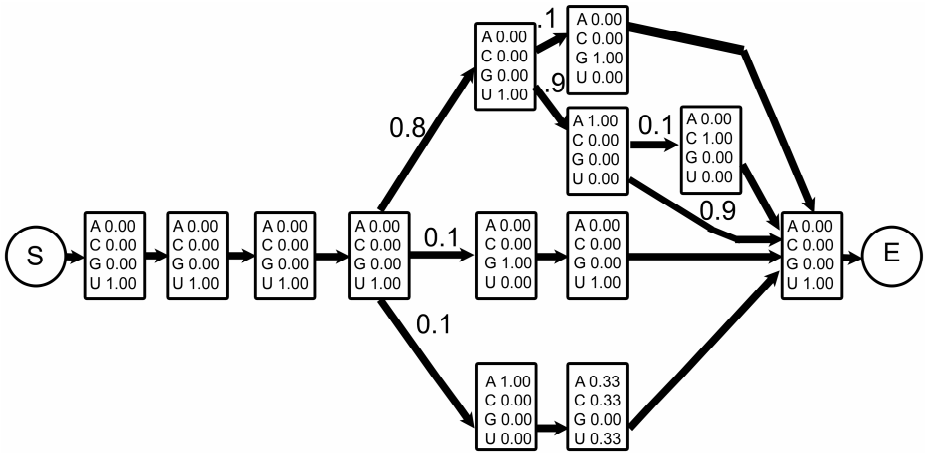


Fig. 2. Description of FSA used for evaluating sequences for CPE sites. Emission probabilities of each of the four nucleotide bases are listed inside the boxes. Transition probabilities are 1, unless otherwise labeled.

2.2 PAS Modeling

For each input sequence, potential PAS regions are detected using an ungapped six-state HMM. For each of the six match states, emission probabilities are constructed according to the weighted frequency of each of the four nucleotides A, C, G, and T/U found in functional poly (A) sites at the corresponding hexamer position, according to the observed frequencies reported in Table 1. Transitions between each of these are set to 1, allowing only for matches. Fig. 3 shows an illustration of the poly(A) hexamer model.



Fig. 3. Description of FSA used for evaluating sequences for PAS hexamer sequences. Emission probabilities of each of the four nucleotide bases are listed inside the boxes. All transition probabilities are set to one.

2.3 GAP Modeling

Mutagenesis studies of cyclin B1-B5 mRNAs indicate that the distance between multiple CPE sites and between the CPE and HEX site are critical in determining translational control by CPEB [31]. Based on these studies, the optimal length between CPE elements has been determined as 10-12 nts, while greater distances produce very weak translational associations. In addition, an optimal distance between the CPE and HEX site is 6-25 nts, while a distance greater than 120 nts is shown to be nonfunctional. Using the information gained through these mutagenesis studies, two

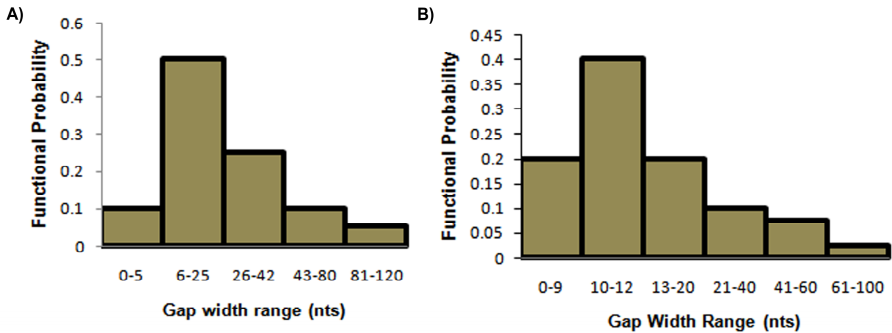


Fig. 4. Probability distributions for CPE-HEX (Fig. A) and CPE-CPE (Fig. B) gaps. Longer and shorter gaps are given a probability value of zero.

duration gap models are created using a binned approach whereby gaps of specified length ranges are assigned probabilities. One probabilistic model is used for CPE-CPE distance, while the second model is used for CPE-HEX distance. Fig. 4 shows the probability distribution for both of these gap models.

After the locations for positive scoring CPE and HEX elements are located for each input sequence, the sequence is scored according to three values: 1) the maximum score for a single occurrence of CPE; 2) the maximum score for two CPEs; and 3) the maximum score for three CPEs. These values (and the corresponding CPE and HEX locations) are stored and reported to the user. This algorithm runs in $O(l + mm^3)$ time where l is the length of the input sequence; m is the number of HEX sites detected and n is the number of CPE sites detected.

3 Results

In order to test the effectiveness of this approach, 3' UTR sequences were downloaded from the UTRdb portion of UTRresource [32–34] dated 6/16/2006. UTRdb contains both 5' and 3' UTRs for a number of different organisms. For the purpose of our study, we downloaded the 3'UTRs for the mouse genome. After finding the results for the mouse genome, we determined the GO enriched datasets using Ontologizer 2.0 [35,36].

A total of 18,342 nonredundant 3' UTRs were extracted from the mouse dataset. Of these, 10,150 show at least a subtle signal for a single CPE element while 4,127 of these show potential for two CPEs, and 1,514 show potential for three CPEs. Using as an input into Ontologizer 2.0 the Entrez gene identifiers for those with three CPEs, a total of 54 different gene ontology identifiers have significant enrichment in this dataset, include 20 under “biological function” subgroup (Table 3). Examination of these 20 GO categories shows a large number involved in development, morphogenesis, and cell signaling, which is in line with the observed role of CPEB regulation.

As an additional measure, 27 mouse UTRs shown to be functionally regulated by CPEB [31] were examined by CPEDetector. All 27 had at least one potential CPE site, with five showing three potential sites and an additional nine showing two

Table 3. Enriched Gene Ontology Biological Function Groups

GO ID	Name
GO:0007154	Cell communication
GO:0032501	Multicellular organismal process
GO:0065007	Biological regulation
GO:0032502	Developmental process
GO:0016043	Cellular component organization
GO:0007399	Nervous system development
GO:0009653	Anatomical structure morphogenesis
GO:0048856	Anatomical structure development
GO:0048869	Cellular developmental process
GO:0009987	Cellular process
GO:0008150	Biological process
GO:0007165	Signal transduction
GO:0048731	System development
GO:0030035	Microspike biogenesis
GO:0030031	Cell projection biogenesis
GO:0050789	Regulation of biological process
GO:0050794	Regulation of cellular process
GO:0007275	Multicellular organismal development
GO:0000902	Cell morphogenesis
GO:0032989	Cellular structure morphogenesis

potential sites. Piqué et al. describe a simplistic regular expression detection routine which detects potential motifs for CPE regulation in 31% of all mouse 3' UTRs [31]. CPEDetector is able to detect a number of additional potential CPE sites (results not shown). CPEDetector has the additional feature of providing a score for each 3' UTR sequence which can be useful in determining which genes should be further studied for CPEB regulation.

4 Discussion

Perhaps the biggest drawback to CPEDetector is the lack of available data regarding sequences known to be regulated by CPEB. This makes it extremely difficult to accurately model CPE motif sites and gaps between CPE motifs and the HEX site. CPEDetector currently takes into account the best estimate of these parameters. It is hoped that by using CPEDetector as a first pass, additional genes may be confirmed to be regulated by CPEB. If this is achieved, then an appropriate training and testing set can be constructed. Due to the current lack of a training set, measures for sensitivity and specificity are unavailable. Therefore, CPEDetector should be used much in the lines of current homology detection algorithms such as BLAST [37] as an initial filter for further biological investigation.

Acknowledgments. The authors would like to thank the organizers of the 1st International Conference on Bioinformatics and Computational Biology for considering our paper and for the anonymous reviewers. Support for this project was provided by

NIH-NCRR grant P20RR16481, NIH-NIEHS grant P30ES014443, and NIH-NEI grant R01EY017594. The contents of this manuscript are solely the responsibility of the authors and may not represent the official views of the National Center for Research Resources, the National Institute for Environmental and Health Science, the National Eye Institute, or the National Institutes of Health. The authors would like to thank the members of the University of Louisville Bioinformatics Research Group (BRG) and the University of Louisville Bioinformatics Laboratory for numerous fruitful discussions.

References

1. Beadoing, E., Freier, S., Wyatt, J.R., Claverie, J.M., Gautheret, D.: Patterns of Variant Polyadenylation Signal Usage in Human Genes. *Genome Res.* 10, 1001–1010 (2000)
2. Clegg, K.B., Piko, L.: RNA Synthesis and Cytoplasmic Polyadenylation in the One-Cell Mouse Embryo. *Nature* 295, 343–344 (1982)
3. Clegg, K.B., Piko, L.: Poly(A) Length, Cytoplasmic Adenylation and Synthesis of Poly(A)+ RNA in Early Mouse Embryos. *Dev. Biol.* 95, 331–341 (1983)
4. Clegg, K.B., Piko, L.: Quantitative Aspects of RNA Synthesis and Polyadenylation in 1-Cell and 2-Cell Mouse Embryos. *J. Embryol. Exp. Morphol.* 74, 169–182 (1983)
5. Wilt, F.H.: Polyadenylation of Maternal RNA of Sea Urchin Eggs After Fertilization. *Proc. Natl. Acad. Sci. U. S. A.* 70(8), 2345–2349 (1973)
6. Richter, J.D.: Translational control during early development. *Bioessays* 13, 179–183 (1991)
7. Simon, R., Tassan, J.P., Richter, J.D.: Translational Control by Poly(A) Elongation During *Xenopus* Development: Differential Repression and Enhancement by a Novel Cytoplasmic Polyadenylation Element. *Genes Dev.* 6, 2580–2591 (1992)
8. Fox, C.A., Sheets, M.D., Wickens, M.P.: Poly(A) Addition During Maturation of Frog Oocytes: Distinct Nuclear and Cytoplasmic Activities and Regulation by the Sequence UUUUUAU. *Genes Dev.* 3, 2151–2162 (1989)
9. McGrew, L.L., Dworkin-Rastl, E., Dworkin, M.B., Richter, J.D.: Poly(A) Elongation During *Xenopus* Oocyte Maturation Is Required for Translational Recruitment and Is Mediated by a Short Sequence Element. *Genes Dev.* 3, 803–815 (1989)
10. McGrew, L.L., Richter, J.D.: Translational Control by Cytoplasmic Polyadenylation During *Xenopus* Oocyte Maturation: Characterization of Cis and Trans Elements and Regulation by Cyclin/MPF. *EMBO J.* 9, 3743–3751 (1990)
11. Paris, J., Swenson, K., Piwnica-Worms, H., Richter, J.D.: Maturation-Specific Polyadenylation: in Vitro Activation by P34cdc2 and Phosphorylation of a 58-KD CPE-Binding Protein. *Genes Dev.* 5, 1697–1708 (1991)
12. Chang, J.S., Tan, L., Schedl, P.: The *Drosophila* CPEB Homolog, Orb, Is Required for Oskar Protein Expression in Oocytes. *Dev. Biol.* 215, 91–106 (1999)
13. Bally-Cuif, L., Schatz, W.J., Ho, R.K.: Characterization of the Zebrafish Orb/CPEB-Related RNA Binding Protein and Localization of Maternal Components in the Zebrafish Oocyte. *Mech. Dev.* 77, 31–47 (1998)
14. Luitjens, C., Gallegos, M., Kraemer, B., Kimble, J., Wickens, M.: CPEB Proteins Control Two Key Steps in Spermatogenesis in *C. Elegans*. *Genes Dev.* 14, 2596–2609 (2000)
15. Walker, J., Minshall, N., Hake, L., Richter, J., Standart, N.: The Clam 3' UTR Masking Element-Binding Protein P82 Is a Member of the CPEB Family. *RNA* 5, 14–26 (1999)

16. Si, K., Giustetto, M., Etkin, A., Hsu, R., Janisiewicz, A.M., Miniaci, M.C., Kim, J.H., Zhu, H., Kandel, E.R.: A Neuronal Isoform of CPEB Regulates Local Protein Synthesis and Stabilizes Synapse-Specific Long-Term Facilitation in *Aplysia*. *Cell*. 115, 893–904 (2003)
17. Stebbins-Boaz, B., Cao, Q., de Moor, C.H., Mendez, R., Richter, J.D.: Maskin Is a CPEB-Associated Factor That Transiently Interacts With EIF-4E. *Mol. Cell*. 4, 1017–1027 (1999)
18. Richter, J.D.: Cytoplasmic Polyadenylation in Development and Beyond. *Microbiol. Mol. Biol. Rev.* 63, 446–456 (1999)
19. Richter, J.D.: CPEB: a Life in Translation. *Trends Biochem. Sci.* 32, 279–285 (2007)
20. Richter, J.D.: Breaking the Code of Polyadenylation-Induced Translation. *Cell* 132, 335–337 (2008)
21. 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, 208–214 (1993)
22. Neuwald, A.F., Liu, J.S., Lawrence, C.E.: Gibbs Motif Sampling: Detection of Bacterial Outer Membrane Protein Repeats. *Protein Sci.* 4, 1618–1632 (1995)
23. Thompson, W., Rouchka, E.C., Lawrence, C.E.: Gibbs Recursive Sampler: Finding Transcription Factor Binding Sites. *Nucleic Acids Res.* 31, 3580–3585 (2003)
24. Stormo, G.D., Hartzell III, G.W.: Identifying Protein-Binding Sites From Unaligned DNA Fragments. *Proc. Natl. Acad. Sci U. S. A.* 86, 1183–1187 (1989)
25. Bateman, A., Haft, D.H.: HMM-Based Databases in InterPro. *Brief. Bioinform.* 3, 236–245 (2002)
26. Sonnhammer, E.L., Eddy, S.R., Durbin, R.: Pfam: a Comprehensive Database of Protein Domain Families Based on Seed Alignments. *Proteins* 28, 405–420 (1997)
27. Burge, C.B., Karlin, S.: Finding the Genes in Genomic DNA. *Curr. Opin. Struct. Biol.* 8, 346–354 (1998)
28. Korf, I., Flicek, P., Duan, D., Brent, M.R.: Integrating Genomic Homology into Gene Structure Prediction. *Bioinformatics* 17(suppl. 1), S140–S148 (2001)
29. Lowe, T.M., Eddy, S.R.: TRNAscan-SE: a Program for Improved Detection of Transfer RNA Genes in Genomic Sequence. *Nucleic Acids Res.* 25, 955–964 (1997)
30. Lowe, T.M., Eddy, S.R.: A Computational Screen for Methylation Guide SnoRNAs in Yeast. *Science* 283, 1168–1171 (1999)
31. Pique, M., Lopez, J.M., Foissac, S., Guigo, R., Mendez, R.: A Combinatorial Code for CPE-Mediated Translational Control. *Cell* 132, 434–448 (2008)
32. UTRResource, <http://www.ba.itb.cnr.it/UTR/>
33. Mignone, F., Grillo, G., Licciulli, F., Iacono, M., Liuni, S., Kersey, P.J., Duarte, J., Saccone, C., Pesole, G.: UTRdb and UTRsite: a Collection of Sequences and Regulatory Motifs of the Untranslated Regions of Eukaryotic MRNAs. *Nucleic Acids Res.* 33, D141–D146 (2005)
34. Pesole, G., Liuni, S., Grillo, G., Saccone, C.: UTRdb: a Specialized Database of 5'- and 3'-Untranslated Regions of Eukaryotic MRNAs. *Nucleic Acids Res.* 26, 192–195 (1998)
35. Ontologizer2.0, <http://compbio.charite.de/index.php/ontologizer2.html>
36. Bauer, S., Grossmann, S., Vingron, M., Robinson, P.N.: Ontologizer 2.0—a Multifunctional Tool for GO Term Enrichment Analysis and Data Exploration. *Bioinformatics* 24, 1650–1651 (2008)
37. 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)

Multiple Sequence Alignment System for Pyrosequencing Reads

Fahad Saeed¹, Ashfaq Khokhar¹, Osvaldo Zagordi², and Niko Beerenwinkel²

¹ Department of Electrical and Computer Engineering,
University of Illinois at Chicago, IL USA

² Department of Biosystems Science and Engineering
ETH Zurich, Basel, Switzerland

Abstract. Pyrosequencing is among the emerging sequencing techniques, capable of generating upto 100,000 overlapping reads in a single run. This technique is much faster and cheaper than the existing state of the art sequencing technique such as Sanger. However, the reads generated by pyrosequencing are short in size and contain numerous errors. In order to use these reads for any subsequent analysis, the reads must be aligned. Existing multiple sequence alignment methods cannot be used as they do not take into account the specific positions of the sequences with respect to the genome, and are highly inefficient for large number of sequences. Therefore, the common practice has been to use either simple pairwise alignment despite its poor accuracy for error prone pyroreads, or use computationally expensive techniques based on sequential gap propagation. In this paper, we develop a computationally efficient method based on domain decomposition, referred to as *pyro-align*, to align such large number of reads. The proposed alignment algorithm accurately aligns the erroneous reads in a short period of time, which is orders of magnitude faster than any existing method. The accuracy of the alignment is confirmed from the consensus obtained from the multiple alignments.

1 Introduction

Pyrosequencing is among the emerging sequencing techniques developed for determining the sequences of DNA bases from a genome. It is capable of generating up to 100,000 overlapping reads in a single run. However, multitude of factors, such as relatively short read lengths (i.e., as of 2008 an average of 100 – 250 nt compared to 800 – 1000 nt for Sanger sequencing), and limited accuracy of individual reads for repetitive DNA, particularly in the case of monopolymer repeats, present many computational challenges [15] to make pyrosequencing useful for biology and bioinformatics applications.

For over more than a decade, Sanger sequencing has been the cornerstone of genome sequencing including that of microbial genomes. Improvements in DNA sequencing techniques and the advances in data storage and analysis, as well as developments in bioinformatics have reduced the cost to a mere 8000\$ – 10000\$ per megabase of high quality genome draft sequence. However, the need of more efficient and cost effective approaches has led to development of new sequencing technologies such as the 454

GS20 sequencing platform. It is a non-cloning pyrosequencing based platform that is several orders of magnitude faster than the Sanger machines. However, the new technology despite its enormous advantage in terms of time and money will not be able to replace the current Sanger technology, unless the reads generated are properly aligned with respect to the reference genome.

The key issues associated with the use of pyrosequencing technique are as under:

Read Length: The read length is expected to be of the order of 100 – 250bp on average. This is much shorter than the other state of the art Sanger machines which give out consistent read lengths of the order of > 800 – 900bp.

Orientation: This is generally the case for most of the sequencing technologies. Each DNA helix will be broken into the original and its Watson-Crick complement. These would be further broken up into pieces, and there is generally no way to reveal which of the two is it. The problem is more severe and usually encountered for genome reconstruction.

Errors: Each individual DNA sequence or read is likely to have errors in the form of insertions and deletions. It may also have mutations and the pyrosequencer may itself make errors. These errors correspond to homopolymer effects, including extension (insertions), incomplete extensions (deletions), and carry forward errors (insertions and substitutions). Insertions are considered the most common type of error (36% of errors) followed by deletions (27%), ambiguous bases, Ns (21%), and substitutions (16%) [29].

For most practical purposes, pyroreads without any post processing are of limited use. One of the most widely required tasks as a pre processing step for many applications, including haplotype reconstruction [12] [13], analysis of microbial community analysis [3], analysis of genes for diseases [2], is the alignment of these reads with the wild type. For important applications such as viral population estimation or haplotype reconstruction of various viruses e.g., HIV in a population, scientists usually have the

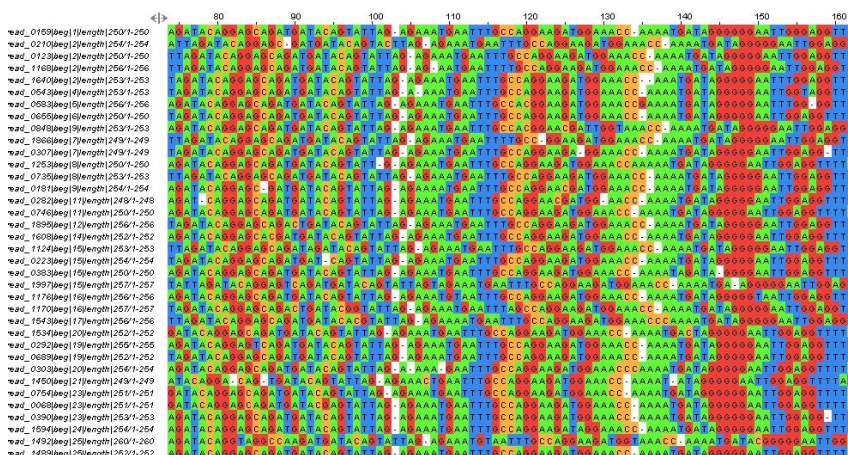


Fig. 1. Pairwise alignment of the reads with the reference genome is shown

information about the wild type genome of the virus. While for other sequencing technologies, such as Sanger, simple pair-wise alignment with the wild type may produce reasonable multiple alignment, in the case of pyrosequencing, the variation in the haplotype population compounded with the errors introduced in the reads does not allow feasible multiple alignment by simple pair-wise alignment. Fig. 1 depicts simple pair-wise alignment of pyrosequence reads with a reference genome. We assert that accurate and workable multiple alignment is often necessary for a variety of applications and statistical packages to work with these pyroreads, as demonstrated in [12] [13] [3] [2].

In theory, alignment of multiple sequences can be achieved using pair-wise alignment, each pair getting alignment score. But for optimal alignment the sum of all the pair-wise alignment scores need to be maximized, which is an NP complete problem [16]. Towards this end, dynamic programming based solutions of $O(L^N)$ complexity have been pursued, where N is the number of sequences and L is the average length of a sequence. Such accurate optimizations are not practical for large number of sequences -as is the case in pyrosequencing-, thus making heuristic algorithms as the only feasible option. The literature on these heuristics is vast and includes widely used works, such as Notredame et. al. [17], Edgar [19], Thompson et. al. [18], Do et. al. [23], and Morgenstern et.al. [21]. These heuristics are complex combination of ad-hoc procedures with some flavor of dynamic programming. Despite the usefulness of these widely used heuristics, they scale very poorly with increasing number of sequences.

For multiple alignment of pyroreads, 'out of the box' use of these heuristics is not feasible because of two main reasons: 1) the pyrosequencing reads can be very large in number (up to 1 million of usable reads in a single run of Roche/454, and 2) the heuristics do not take into account the positions of the reads with respect to the reference genome. Additional factors such as short lengths and errors, and the fact that these reads have preceding or trailing 'gaps' pose further alignment challenges. In [14], an alignment technique based on sequential gap propagation has been used. This technique is computationally expensive and its alignment quality decreases with the increase in the mutation value.

In this paper, we present a computationally efficient algorithm *pyro-align*, specifically designed for multiple alignment of DNA reads obtained from pyrosequencing. The proposed algorithm is based on a novel domain decomposition concept, therefore it is capable of aligning very large number of pyrosequences. It takes into account the position of the reads with respect to the reference genome, and assigns weight to the leading and trailing gaps for the reads.

The objective of our work is to develop a multiple alignment system for small error prone reads, such that the errors in the alignment are 'highlighted' and the system is able to handle large number of reads, as may be expected from pyrosequencing reads.

We assume that the reads may be generated from one or many genomes, with 'forward' orientation. We also assume that the reference genome (or its wild type) from which the reads are generated is available, as is generally the case for haplotype reconstruction. In our experiments, we have used HIV-pol gene virus as the reference genome (with length of 1970bp) and simulator Readsimsim [11] to generate these reads. The algorithm uses concepts from domain decomposition and parallel multiple alignment techniques [1,22].

For the sake of completeness, let's first formally define the Multiple Sequences Alignment problem in its generic form, without indulging with the issues such as scoring functions. Let N sequences be presented as a set $S = \{S_1, S_2, S_3, \dots, S_N\}$ and let $S' = \{S'_1, S'_2, S'_3, \dots, S'_N\}$ be the aligned sequence set, such that all the sequences in S' are of equal length, have maximum overlap, and the score of the global map is maximum according to some scoring mechanism suitable for the application.

A perfect multiple alignment for pyroreads would be, that the reads are aligned with each other such that the position of the reads with respect to the reference genome is conserved; the reads have maximum overlap and are of equal lengths after the alignment, including leading and trailing gaps.

The intuitive idea behind the proposed *pyro-align* algorithm is to first place the reads in correct orientation and position with respect to the reference genome and then use progressive alignment to achieve the final alignment. For efficient progressive alignment, the correctly placed reads are reordered according to the starting position, and a computationally low complexity similarity metric is extracted from this ordering position. The similarity metric is then used to align pairs of aligned reads using a hierarchical decomposition strategy. The proposed multiple alignment algorithm takes advantage of the pyroreads characteristics and brings in techniques from data structures and parallel computing to realize a low complexity solution in terms of time and memory.

The proposed alignment algorithm, *pyro-align*, consists of the following two main components:

1. Semi-Global alignment
2. Hierarchical progressive alignment
 - (a) Reordering of reads to generate guidance tree
 - (b) Pairwise and profile-profile alignment

Each component is designed considering the characteristics of pyroreads and it is described in the following sections along with its justification.

1.1 Semi-global Alignment

The first step is to determine the position of each read with respect to the reference genome. If this step is omitted, there are number of alignments that would be correct, but would be inaccurate if analyzed in the global context. A read that is not constricted in terms of position, may give the same score (SP score) for the multiple alignment but would be incorrect in context of the reference. To accomplish the task of 'placing' the reads in the correct context with respect to the reference genome we employ semi-global alignment procedure.

The semi global alignment is also referred to as overlapping alignment because the sequences are globally aligned ignoring the start and end gaps. For semi-global alignment we use a modified version of Needleman-Wunsch algorithm [5].

The modification in the basic version of Needleman-Wunsch is required to handle the leading and trailing gaps of the reads when aligning to the reference genome. If the leading and trailing gaps are not ignored, considering the short length of the reads, the alignment scores would be dominated by these gaps, hence giving an inaccurate alignment with respect to the genome.

Let the two sequences to be aligned be s and t , and $M(i, j)$ presents the score of the optimal alignment. Since, we do not wish to penalize the starting gaps, we modify the dynamic programming matrix by initializing the first row and first column to be zero. The gaps at the end are also not to be penalized. Let $M(i, j)$ represent the optimal score of s_1, \dots, s_i and t_1, \dots, t_j . Then $M(m, j)$ is the score that represents optimally aligning s with $t_{1, \dots, j}$. The optimal alignment therefore, is now detected as the maximum value on the last row or column. Therefore the best score is $M(i, j) = \max_{k, l}(M(k, n), M(m, l))$, and the alignment can be obtained by tracking the path from $M(i, j)$ to $M(0, 0)$. For additional details on semi-global alignment we refer the reader to [8].

Once each read has been semi-globally aligned with the reference genome, we obtain reads with leading and trailing gaps, where the first character after the gaps is the starting position of the read with respect to the reference genome. The information for these alignments are stored in hashtables that are further used for processing in reordering the reads for alignment.

2 Hierarchical Progressive Alignment

Generally multiple sequence alignment (MSA) procedures are either based on iterative methods or employ progressive techniques. Although progressive techniques relative to iterative techniques are more efficient, they are not suitable when the sequences are relatively diverse or the number of sequences is very large. Considering the fact that the pyroreads are highly similar, we develop a hierarchical progressive alignment procedure that is also computationally efficient for large number of reads.

Progressive alignment techniques develop final MSA by combining pair-wise alignments beginning with the most similar pair and progressing to the most distantly related. All progressive alignment methods require two stages: a first stage in which the relationships between the sequences are represented as a tree, called a guide tree, and a second step in which MSA is built by adding the sequences sequentially to the growing MSA according to the guide tree. In the following, we describe the low complexity components of *pyro-align*.

2.1 Reordering Reads

The method followed by most of the progressive multiple alignment algorithms is that a quick similarity measure is computed that is based on k-mer counting [4] or some other heuristic mechanism. These pair-wise similarity measures (distances) are tabulated in a matrix form and a tree is constructed from this distance matrix using UPGMA or neighboring joining. The progressive alignment is thus built, following the branching order of the tree, giving a multiple alignment. These steps require $O(N^2)$ time each, where N is the number of reads. To reduce this complexity, we exploit the fact that the reads are coming from the same reference or nearly the same reference. This in turn implies that the reads starting from the same or near same 'starting' point with respect to the reference genome are likely to be similar to each other. Therefore, we already have the

ordering information or the 'guide tree' from the first step of the algorithm. Our guide tree, or the order in which sequences will be aligned in the progressive alignment is from the starting position of the reads obtained in the first stage. Of course the decomposition of the reads (the subtree of the profiles that we built) doesn't render the reads in the same order as in traditional progressive alignment, but nevertheless the order is more or less the same when the profiles of these reads are aligned.

Let there be N number of reads $R = R_1, R_2, \dots, R_N$ generated from pyrosequencing technique, from the reference genome of length L_g . Also, let the length of each read denoted by $L(R)_p$. After executing semi-global alignment using the algorithm discussed in the previous section, let each read be presented by R_{pq} , where the p^{th} read has q leading gaps and at most $L_g - q - L(R)_p$ trailing gaps. Then the reordering algorithm would reorder the reads such that after the reads are reordered using the information from the leading gaps, the read R_{pq} comes in ordering 'before' $R_{p(q+1)}$, $\forall p, q \in L_g$.

To execute the reordering in an efficient manner, we employ hashtables that speed up the search process. We create two hashtables: *hashtable*₁ uses fasta sequence tag as the hash key and stores the corresponding starting position of the read; *hashtable*₂ stores the read names (fasta sequence tag) and the dna sequence it is associated with. Using these tables, the reads are reordered in the database in linear time.

2.2 Pair-Wise and Profile-Profile Alignments

The ordering of the reads determined in the preceding step is now used to conduct the progressive alignment. Traditional progressive alignment requires that the sequences most similar to each other are aligned first. Thereafter, sequences are added one by one to the multiple alignments determined according to some similarity metric. This sequential addition of sequences for progressive alignment is not suitable for large number of sequences. In order to devise a low complexity system, we design a hierarchical progressive alignment procedure that is based on domain decomposition [1], as described below and depicted in Figure 2.

First of all, pair-wise local alignment using standard Needle-Wunsch is executed on each overlapping pair of reads (the ordering is still the same as discussed in the previous section). After this stage, the reads are aligned in pairs such that we have $N/2$ pairs of aligned reads. These $N/2$ pairs of reads are then used for profile alignments as discussed below.

Profile-profile alignments are used to re-align two or more existing alignments (in our case the pairs of aligned reads). It is useful for two reasons; one being that the user may want to add sequences gradually, and second being that the user may want to keep one high quality profile fixed and keep on adding sequences aligned to that fixed profile [18].

We take advantage of both of these properties in our domain decomposition.

In this stage of the algorithm, the $N/2$ pairs of aligned reads have to be combined to get a multiple alignment. We have shown in [22] that the decomposition of the profiles gives a fair amount of time advantages even on a single processor. Therefore a hierarchical model similar to [1] is implemented (see Fig. 2). The model requires that instead

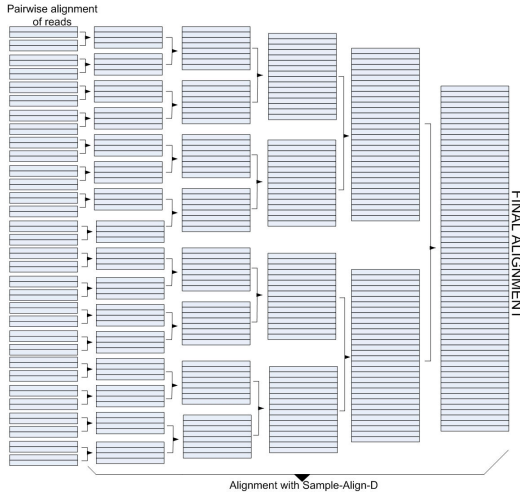


Fig. 2. Hierarchical profile-profile alignments for pyro-align is shown

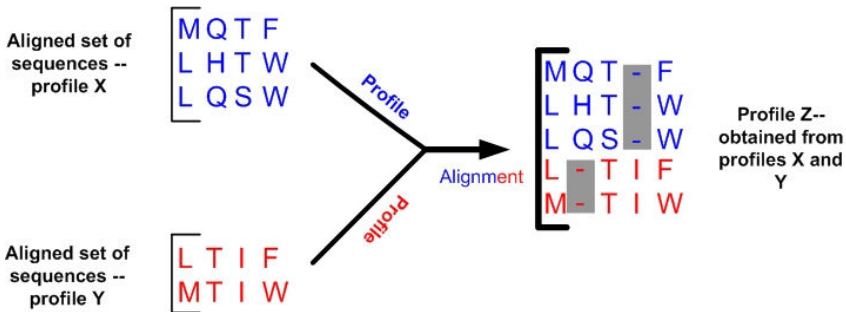


Fig. 3. Two profiles(X and Y) are aligned under the columns constrains, producing profile Z

of combining the profiles in a sequential manner (one by one), a binary tree is built such that the profiles to be aligned are the leaves of the tree.

In order to apply pair-wise alignment functions to profiles, a scoring function must be defined, similar to the substitution methods defined for pair-wise alignments. One of the most commonly used profile functions is the sequence-weighted sum of substitution matrix scores for each pair of amino acid letters. Let i and j be the amino acid, p_i the background probability of i , p_{ij} the joint probability of i and j aligned to each other, S_{ij} the substitution matrix being used, f_i^x the observed frequency of i in column x of the first profile, x_G the observed frequency of gaps in that column. The same attributes are assumed for the profile y . Profile sum of pairs (PSP) is the function used in Clustalw [18], Mafft [24] and Muscle [20] to maximize Sum of Pairs(SP) score, which in turn maximizes the alignment score such that the columns in the profiles are preserved, as depicted in Fig. 3. The PSP score can be defined as in [25] and [20]:

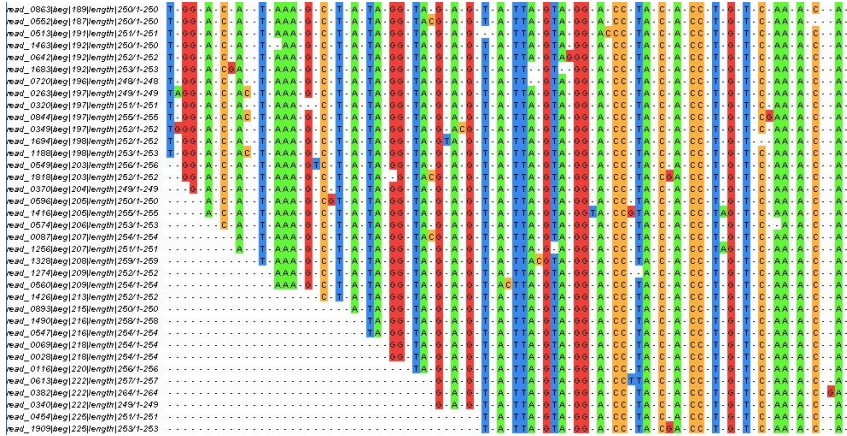


Fig. 4. The final Alignment of the reads

$$PSP^{xy} = \sum_i \sum_j f_i^x f_j^y \log(p_{ij}/p_i p_j) \quad (1)$$

For our purposes, we will take advantage of PSP functions based on 200 PAM matrix [26] and the 240 PAM VTML matrix [27]. Some multiple alignment methods implement different scoring functions such as Log expectation (LE) functions, but for our purposes PSP scoring suffices. Profile functions have evolved to be quite complex and good discussion on these can be found at [20] and [28]. We use the profile functions from the clustalw system. The final alignment from the *pyro-align* algorithm can be seen in Fig. 4. Different steps of the proposed *pyro-align* Algorithm are outlined below.

Algorithm 1. Steps of the Proposed Multiple Sequence Alignment *pyro-align* Algorithm

Input: Reads generated from pyrosequencing procedure and Reference Genome
Output: A Multiple Alignment of Reads is returned
 //Calculate overlapping of each of the reads with respect to the reference Genome
for ($i = 1; i \leq N; i++$) **do**
 | Overlapped-Reads \leftarrow Semi-Global-Alignment(R_i , Genome);
end
 Reordered-Reads \leftarrow Reordering(Overlapped-Reads);
 //Pairwise alignment using standard Needle-Wunsch is executed, for pairs of ordered reads;
 Pair-wise-aligned \leftarrow Needle-Wunsch(Reordered-Reads);
 //Profile-profile alignment is obtained using Sample-align-D strategy
 Final-Alignment \leftarrow Profile-Profile-alignment(Pair-wise-aligned);
return Final-Alignment;

3 Performance Analysis

As discussed earlier in the paper, the exact solution for multiple alignment is not feasible and heuristics are employed. Most of these heuristics perform well in practice but there is generally no theoretical justification possible for them [9]. For pyro-align it can be shown that the semi-global alignment of the reads with the reference genome is analogous to center star alignment. The center star alignment is shown to give results within 2-approx of the optimal alignment [9] in worst case and same can be expected from the semi-global alignment of reads with reference genome. The accuracy of the later stages is confirmed by rigorous quality assessment procedure described in the section below.

3.1 Experimental Setup and Quality Assessment

The performance evaluation of the algorithm has been carried on a single desktop system 2x QuadCore Intel 5355 2.66 GHz, 2x4 MB Cache and 16GB of RAM. The operating system on the desktop is RedHat Linux with kernel 2.6.18-92.1.13.el5. The software uses libraries from Biojava [7] and is built using java version "1.6.0" Java(TM) SE Runtime Environment, IBM J9 VM.

To investigate the quality of the alignment produced by the algorithm we used Readsims simulator [11] to generate the reads. The quality assessment of multiple alignment is generally carried out using benchmarks such as Prefab [19] or BaliBase [6]. However, these benchmarks are not designed to assess the quality of the aligned reads produced from pyrosequencing, and there are no benchmarks available specifically for these reads. Therefore, a system has to be developed to assess the quality of the aligned reads. The experimental setup for the quality assessment of the alignment procedure is shown in the Fig. 5 and is explained below.

Our quality assessment has two objectives: (1) to assess the quality of the alignment produced by pyro-align with respect to the original genome (2) ensure that the system must be able to handle reads from multiple haplotype for alignment.

To achieve these objectives, we setup the quality assessment system as shown Fig. 5. We used a HIV pol gene virus with length of 1970bp as the wildtype for the experiments. The wildtype is then used to produce 4 genomes, randomly mutated at different rate; The four sets of genomes are Dist-003, Dist-005, Dist-007 and Dist-010, with mutations of 3%, 5%, 7% and 10%, respectively. Now using the mutated genomes, 2000 and 5000 reads from the Readsims were generated using standard ReadSim parameters with forward orientation.

The generated reads from these mutated genomes were then aligned with the wildtype. This procedure is adopted because generally scientists only have a wildtype sequence of the microbial genomes available and therefore it depicts a more practical scenario.

After the alignment, a majority consensus of the reads is obtained. A distance based similarity is then calculated of the consensus obtained from the aligned reads with the original genome from which the reads were generated. The results of the alignment obtained and the accuracy of the consensus thus obtained are shown in Fig. 6 and Fig. 7 for 2000 and 5000 reads respectively.

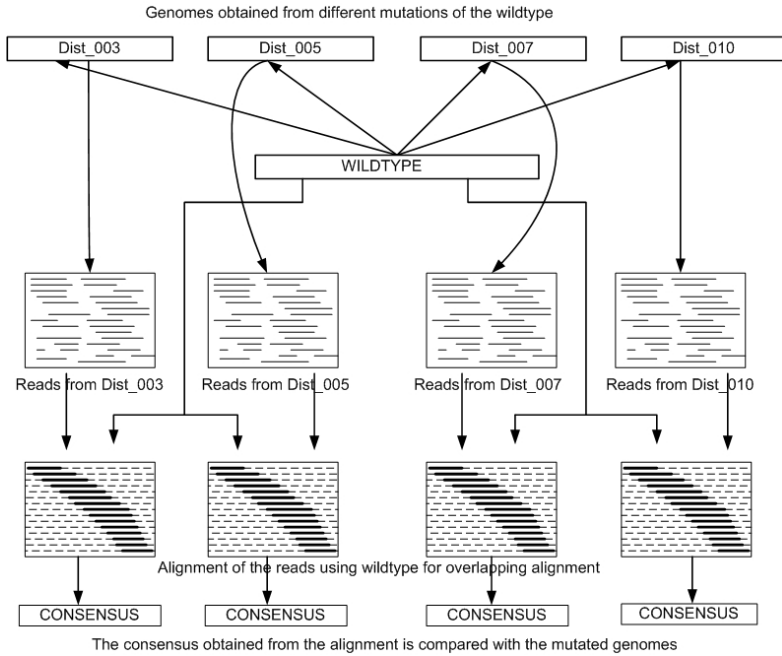


Fig. 5. The experimental setup for the quality assessment of the multiple alignment program

We compare the accuracy of the algorithm with two different methods. First being the simple pair-wise alignment of the reads with the reference genome. Secondly, we compare it with a sequential gap propagation method, used in recent pyrosequencing systems [14]. Simply put, gap propagation method builds multiple alignment from pairwise alignments by sequentially 'propagating' the gaps from each pairwise alignment to all the reads in the system. Propagation of gaps is accomplished for every position where at least one read has an inserted base. A gap is inserted in the reference genome and, consequently, in all reads that overlap the genome at that position. The complexity of the procedure is of the order of $O(N^2)$.

The accuracy of the consensus obtained using just the pairwise alignment is less than 55% and that obtained from the pyro-align is always greater than 96%. An even better alignment quality is achieved for greater number of reads, because more number of reads provide a better coverage for a genome of given length. The accuracy of the gap propagation procedure, is comparable to pyro-align for small mutations, but as the mutations increase the accuracy of gap propagation based method decreases.

To illustrate that the alignment system also works with a 'mixture' of reads from different haplotype, we use the mutated reads from Dist-003, Dist-005 and Dist-007 to generate a new set of reads. The new set contains equal number of reads from the mutated sets e.g. 2000 reads from each mutated genome for the results shown. The reads are then aligned by the pyro-align algorithm using wildtype as the reference genome. The results of alignment for this mixture set are shown in Fig. 8 for Dist-003/Dist-005 and Dist-005/Dist-007 mixtures. It must be noted here that we don't have a 'ground

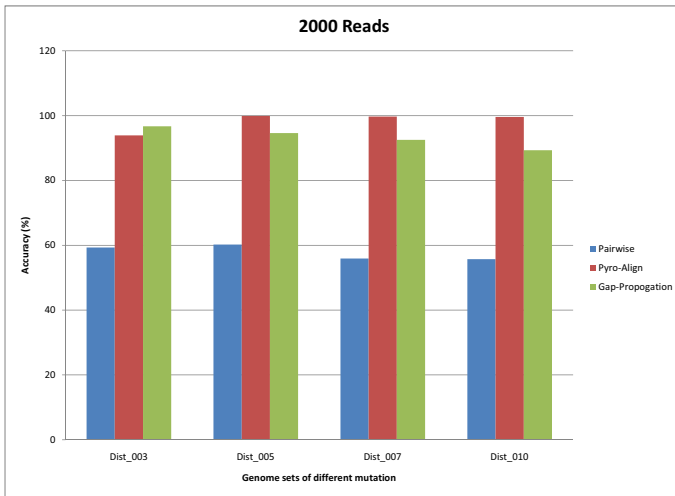


Fig. 6. The quality of the alignment using pairwise, pyro-Align and 'propagation' methods for 2000 reads



Fig. 7. The quality of the alignment using pairwise, pyro-Align and 'propagation' methods for 5000 reads

truth' genome in these cases and hence no genome is available to compare the consensus obtained from the alignment.

However, we do know the mutation rates for the genomes from which the mixture sets were generated. Therefore, if an optimal alignment of these reads is obtained, the 'mutation' in the consensus should not be greater than the combined mutations of the genomes. For example consider the case of Dist-003/Dist-005 mixture. We know

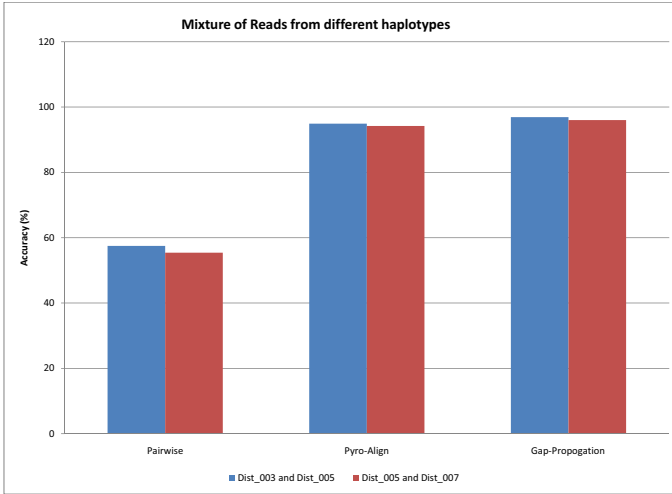


Fig. 8. The quality of the alignment using pairwise, pyro-Align and 'propagation' methods for mixed genome reads

the mutation rates for the genomes from which the reads generated were 3% and 5% with respect to the wildtype. Therefore, for accurate alignment, the consensus of the alignment should not vary more than 8%, in the worst case, when compared to the wildtype. Same would be true for the other cases considered according to the mutation rates of the genomes. As can be seen that the results of the alignment compared with the wildtype are well within the expected limits. The accuracy of the pairwise alignment of the reads with the reference genome (in this case the wildtype), and that obtained using propagation method is also shown for comparison.

4 Complexity Analysis

In this section we briefly outline the complexity of the proposed pyro-align algorithm. Recall the pyro-align algorithm consists of these major steps: semi-global alignment, reordering, pair-wise alignment, and profile-profile alignment.

We assume that the number of reads is N with the average length of the read equal to L_R . Let's further assume that the length of the reference genome is equal to L_g . Then, the complexity of the semi-global alignment (overlapping alignment) is equal to $O(NL_R L_g)$. The clustering of the reads can be done in $O(NL_g)$ and the reordering using hashables can be achieved in $O(N)$, making the total for this stage equal to $O(NL_g + N)$. The pairwise alignment of the reads is shown to be achieved in $O(NL_R^2)$ and the profile-profile alignment can be achieved in $O(N \log N \times L_g^2)$. This makes the total complexity equal to $O(NL_R L_g + NL_g + N + NL_R^2 + N \log N \times L_g^2)$. This is asymptotically equal to $O(N \log N \times L_g^2 + NL_R^2)$. The advantage of low complexity of pyro-align was further evident by our experimentation. We were able to align 2000 reads of average length 250bp from a genome of length 1970bp in about 12 minutes

compared to 6 hours of computation using more traditional multiple alignment systems such as Clustalw.

5 Conclusion

The short reads from the pyrosequencing method are rendered useless if they are not multiple aligned for magnitude of important applications, such as haplotype reconstruction and error elimination. We have presented an efficient hierarchical procedure to multiple align large number of short reads from the pyrosequencing procedure.

We demonstrated that simple-pair-wise alignment is not feasible in the case of pyroreads. We also showed that the proposed method is much faster than traditional time consuming multiple alignment methods such as Clustalw or Tcoffee. We also presented the quality assessment results and compared those with the results obtained by simple pair-wise alignment procedure and 'propagation' methods.

Acknowledgements. The work was done, in part when Fahad Saeed was visiting Biosystems Science and Engineering Department (D-BSSE), ETH Zurich, Switzerland and in part, at Department of Electrical and Computer Engineering, University of Illinois at Chicago. Saeed was additionally supported by ThinkSwiss Scholarship by the government of Switzerland.

References

1. Saeed, F., Khokhar, A.: Sample-Align-D: A High Performance Multiple Sequence Alignment System using Phylogenetic Sampling and Domain Decomposition. In: Proc. 23rd IEEE International Parallel and Distributed Processing Symposium (April 2007)
2. Hou1, X.-L., Cao, Q.-Y., Jia, H.-Y., Chen, Z.: Pyrosequencing analysis of the *gyrB* gene to differentiate bacteria responsible for diarrheal diseases. *European Journal of Clinical Microbiology & Infectious Diseases* 27(7), 587–596 (2007)
3. Liu, Z., Lozupone, C., Hamady, M., Bushman, F.D., Knight, R.: Short pyrosequencing reads suffice for accurate microbial community analysis. *Nucl. Acids Res.* 541 (2007)
4. Edgar, R.C.: Local homology recognition and distance measures in linear time using compressed amino acid alphabets. *Nucl. Acids Res.* 32(1), 380–385 (2004)
5. 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)
6. Thompson, J.D., Plewniak, F., Poch, O.: BALiBASE: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics* 15(1), 87–88 (1999)
7. Pocock, M., Down, T., Hubbard, T.: BioJava: open source components for bioinformatics. *SIGBIO Newsl* 20(2), 10–12 (2000)
8. Setubal, C., Meidanis, J.: *Introduction to Computational Molecular Biology* (January 1997)
9. Gusfield, D.: *Algorithms on Strings, Trees, and Sequences. Computer Science and Computational Biology* (January 1997)
10. Gusfield, D.: Efficient methods for multiple sequence alignment with guaranteed error bounds. Computer Science Division, UC Davis, Technical Report CSE 91-4 (1991)
11. Schmid, R., Schuster, S.C., Steel, M.A., Huson, D.H.: ReadSim-A simulator for Sanger and 454 sequencing (2006)

12. Eriksson, N., Pachter, L., Mitsuya, Y., Rhee, S.-Y., Wang, C., Gharizadeh, B., Ronaghi, M., Shafer, R.W., Beerenwinkel, N.: Viral Population Estimation Using Pyrosequencing: PLoS Comput Biol. Public Library of Science 4 (May 2008)
13. Wang, C., Mitsuya, Y., Gharizadeh, B., Ronaghi, M.: Characterization of mutation spectra with ultra-deep pyrosequencing, application to HIV-1 drug resistance. *Genome Res.* 17(8), 1195–1201 (2007)
14. Zagordi, O., Geyrhofer, L., Roth, V., Beerenwinkel, N.: Deep sequencing of a genetically heterogeneous sample: local haplotype reconstruction and read error correction. In: RECOMB 2009 (accepted paper) (2009)
15. Hutchison III, C.A.: DNA sequencing, bench to bedside and beyond. *Nucleic Acids Research* 35, 6227–6237 (2007)
16. Wang, L., Jiang, T.: On the Complexity of Multiple Sequence Alignment. *Journal of Computational Biology* 1(4), 337–348 (1994)
17. Notredame, C., Higgins, D., Heringa, J.: T-coffee: A novel method for multiple sequence alignments. *Journal of Molecular Biology* 302, 205–217 (2000)
18. Thompson, J., Higgins, D., Gibson, T.J.: Clustal w: improving the sensitivity of progressive multiple alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* 22, 4673–4690 (1994)
19. Edgar, R.C.: MUSCLE: Multiple Sequence Alignment with High Accuracy and High Throughput. *Nucleic Acids Research* 32(5) (2004)
20. Edgar, R.C.: MUSCLE: A Multiple Sequence Alignment Method with Reduced Time and Space Complexity. *BMC Bioinformatics*, 1471–2105 (2004)
21. Morgenstern, B.: DIALIGN: multiple DNA and protein sequence alignment at BiBiServ. *Nucleic Acids Research* 32, 33–36 (2004)
22. Saeed, F., Khokhar, A.: A Domain Decomposition Strategy for Alignment of Multiple Biological Sequences on Multiprocessor Platforms. *Journal of Parallel and Distributed Computing* (to appear)
23. Do, C.B., Mahabhashyam, M.S.P., Brudno, M., Batzoglu, S.: PROBCONS: Probabilistic Consistency-based Multiple Sequence Alignment. *Genome Research* 15, 330–340 (2005)
24. Katoh, K., Misawa, K., Kuma, K., Miyata, T.: MAFFT A Novel Method for Rapid Multiple Sequence Alignment based on Fast Fourier Transform. *Nucleic Acids Res.* 30(14), 3059–3066 (2002)
25. Altschul, S.F.: Amino acid substitution matrices from an information theoretic prospective. *J. Mol. Biol.* 219(3), 555–565 (1991)
26. Jones, D.T., Taylor, W.R., Thornton, J.M.: The rapid generation of mutation data matrices from protein sequences. *BMC Bioinformatics* 8(3), 275–282 (1991)
27. Müller, T., Spang, R., Vingron, M.: Estimating Amino Acid Substitution Models: A Comparison of Dayhoff's Estimator, the Resolvent Approach and a Maximum Likelihood Method. *Mol. Bio. Evol.* 19(1), 8–13 (2002)
28. Edgar, R.C., Sjolander, K.: A comparison of scoring functions for protein sequence profile alignment. *Bioinformatics* 20(8), 1301–1308 (2004)
29. Huse, S., Huber, J., Morrison, H., Sogin, M., Welch, D.: Accuracy and quality of massively parallel DNA pyrosequencing. *Genome Biology* 8(7), R143 (2007)
30. Roche Applied Sciences: GS20 Data Processing Software Manual: Penzberg: Roche Diagnostics GmbH (2006)

A Bayesian Approach to High-Throughput Biological Model Generation

Xinghua Shi¹ and Rick Stevens^{1,2,*}

¹ Department of Computer Science, University of Chicago, Chicago, IL 60637, USA
shi@uchicago.edu

² The Computing, Environment and Life Science, Argonne National Laboratory, Argonne, IL 60439, USA
stevens@anl.gov

Abstract. With the availability of hundreds and soon thousands of complete genomes, the construction of genome-scale metabolic models for these organisms has attracted much attention. Manual work still dominates the process of model generation, however, and leads to the huge gap between the number of complete genomes and genome-scale metabolic models. The challenge in constructing genome-scale models from existing databases is that usually such a directly extracted model is incomplete and contains network holes. Network holes occur when a network is disconnected and certain metabolites cannot be produced or consumed. In order to construct a valid metabolic model, network holes need to be filled by introducing candidate reactions into the network. As a step toward the high-throughput generation of biological models, we propose a Bayesian approach to improving draft genome-scale metabolic models. A collection of 23 types of biological and topological evidence is extracted from the SEED [1], KEGG [2], and BiGG [3] databases. Based on this evidence, we create 23 individual predictors using Bayesian approaches. To combine these individual predictors and unify their predictive results, we build an ensemble of individual predictors on majority vote and four classifiers: naive Bayes classifier, Bayesian network, multilayer perceptron network and AdaBoost. A set of experiments is performed to train and test individual predictors and integrative mechanisms of single predictors and to evaluate the performance of our approach.

1 Introduction

The number of annotated genomes is approaching 1000, thanks to high-throughput sequencing technology in biology and automated genome annotation tools in bioinformatics. The availability of these complete genomes enables analyzing genomes at the system level. One such analysis has been carried out through the construction of a genome-scale metabolic model for a microorganism from its genome sequence. Starting

* We acknowledge federal funds from the National Institute of Allergy and Infectious Diseases, National Institutes of Health, Department of Health and Human Services, under Contract No. HHSN266200400042C. This work was supported in part by the U.S. Dept. of Energy under Contract DE-AC02-06CH11357.

from the extraction of gene-protein-reaction (GPR) associations from genome and pathway databases, one can build genome-scale metabolic models using constraint-based approaches such as flux balance analysis [7,8,10].

However, a genome-scale metabolic model generated by directly extracting GPR associations from existing databases is usually incomplete and contains network holes. Network holes are those places where certain metabolites cannot be produced or consumed as there are no reactions to connect these metabolites. In order to construct a valid model, every metabolite should have fluxes pass through it. Candidate reactions must be introduced to fill network holes. A number of factors can lead to network holes such as missing genes, wrong or missing annotations, and poor mappings from functions to biochemical reactions. At present, manual search for hole-filling candidates dominates the work of filling network holes in the construction of genome-scale metabolic models. Because of the huge volume of data available to act as evidence and the large scale of metabolic networks, manual work is time-consuming and labor-intensive. Up to now, approximately 27 genome-scale metabolic models have been constructed [6]. In comparison, the number of complete genomes is approaching 1000. With the exponential growth of the number of genomes, the gap between the number of genomes and the number of genome-scale metabolic models is expanding. In order to bridge this gap and generate 1000 genome-scale metabolic models in the near future, it is desirable that computational approaches be applied to fill network holes and thus accelerate the model-building process.

2 Related Work

Computational approaches have been proposed to improve metabolic networks or metabolic pathways. Green and Karp [14] showed a Bayesian approach to identify missing enzymes for filling pathway holes in pathway/genome databases by integrating evidence from homology, operon, and metabolic pathway relationships. However, the networks generated from their approach are incomplete at the network level, and network holes remain in the networks.

Kharchenko et al. [15] developed a computational approach for selecting candidate genes that can be assigned to missing metabolic enzymes, based on the gene expression data and structure of partially reconstructed metabolic network. Chen and Vitkup [12] presented a method that uses local structure of a metabolic network combining with phylogenetic profiles to suggest candidate genes for enzymes without corresponding genes. Kharchenko et al. [11] expanded their methods to include multiple types of functional association evidence, including clustering of genes on the chromosome, similarity of phylogenetic profiles, gene expression, protein fusion events, and a local structure of metabolic network to infer genes encoding for a specific metabolic function. However, the collection of algorithms presented in [11,12,15] focuses on filling network holes where a single network hole is present in a neighborhood of metabolic networks. In practice, patches of network or a collection of holes commonly occur in a metabolic network, especially in organisms whose genomes are not well annotated. Therefore, these approaches are not suitable for the massive production of genome-scale metabolic models.

In [13], DeJongh et al. presented tools for the generation of substantially complete metabolic networks for over 400 complete genome sequences currently in the SEED. Their tools are based on the notation of scenarios that represent segments of metabolic pathways with connected reactions accompanied by input and output compound sets. By assembling such scenarios, one can construct an organism-specific reconstruction of the metabolic network. Arguably, the work in [13] enhances the curation of associations between functional roles and reactions and thus generates better GPR associations for the reconstruction of genome-scale metabolic networks in the SEED. However, the reconstructed networks produced by their tools are still partially complete and include many network holes. Therefore, in order to obtain a complete metabolic network and then build a valid model, further work is needed to fill network holes in these draft metabolic networks.

3 Our Approach and Tools

The Rapid Annotation using Subsystem Technology (RAST) server [18] provides rapid and fully automated annotations for bacterial genomes. Users can submit their new genomes, and normally RAST will complete annotations and make the annotated genome available within 12–24 hours. The SEED [1,17] provides an environment and tools that curate function assignments based on subsystems. The work in [13] generates a more accurate reaction set for an annotated genome using the technology of metabolic scenarios tightly coupled with subsystems. The work in [13] also provides hundreds of draft metabolic models that can be further improved.

Following the efforts in [13], we design a set of computational tools that, with high-throughput techniques, will generate complete genome-scale metabolic models. This toolset comprises four main parts: parsers to integrate and reconcile data from different databases, network hole detectors that analyze network connectivity and identify network holes, evidence extractors that mine through integrated data and extract pieces of evidence out of the data, and a set of predictors that use evidence to suggest candidate hole-filling reactions. In this paper, we focus on the latter two sets of tools for extracting evidence and for constructing predictors. The hole-filling mechanism proposed in this paper looks at the genome-scale metabolic model gained from an organism's genome annotations and from mining extensively through databases. By expanding a draft model exhaustively in every direction, we seek to improve genome-scale metabolic models and enable mass production of metabolic models.

Without prior knowledge, a biochemically possible reaction can be assumed to be randomly distributed. In other words, the probability of including any reaction in a model is treated as $\frac{1}{|R|}$ if all biochemically possible reaction in KEGG, R , is considered. We know, however, that the probability of each reaction is highly skewed in different datasets: some reactions happen more often than others. Therefore, the probabilities of reactions should be adjusted after seeing certain datasets. And these adjusted probabilities can be viewed as “priors” to pump into the calculation of adjusted reaction probabilities after seeing other datasets. In this paper, reaction probabilities are adjusted by using Bayes' rule.

3.1 Evidence Extraction

In order to generate a candidate reaction list that can be incorporated to complete a partial metabolic model, a set of evidence should be extracted from available data. As shown in Table 1, 23 pieces of evidence of seven different types are extracted from different data sources. They are (a) reaction priors, (b) the co-occurrence of reaction pairs, (c) segment priors, and (d) the co-occurrence of reaction segment pairs for datasets in two reconstructed BiGG models, *iJR904* and *iSB619*; KEGG reference pathway map; KEGG modules; KEGG organism maps; and all draft models in the SEED; as well as (e) the co-occurrence of gene pairs in all organisms in the SEED, (f) the co-occurrence of gene pairs in the SEED, and (g) the co-occurrence of gene-gene pairs in the same clusters on chromosomes in the SEED.

Table 1. Summary of Evidence

Type	BiGG Models	KEGG Ref Map	KEGG Modules	KEGG Maps	Org	SEED Orgs
(a)	[1] 634	[2] 4,953	[3] 434	[4] 3,324		[5] 1,318
(b)	[6] 893	[7] 5,358	[8] 1,765	[9] 10,178		[10] 3,036
(c)	[11]5,175	[12]36,145	[13]11,607	[14] 237,903		[15] 70,326
(d)	[16]16,421	[17]101,101	[18]36,831	[19]1,027,801		[20] 326,834
(e)	/	/	/	/		[21] 376,880
(f)	/	/	/	/		[22] 139,183
(g)	/	/	/	/		[23] 302,664

Each row in the table is a type of evidence and each column is a dataset. The content of each cell represents the index of an evidence, followed by the number of data points in the dataset. For example, “[1]634” means that for the first ([1]) evidence, which is reaction priors in BiGG models, there are 634 reactions in the dataset. The details of each type of evidence are as follows.

(a) *Reaction Priors*: For any reaction r in the KEGG (all reactions in KEGG are denoted as R), if it has been seen in existing pathway maps with some prior probability, then these priors can be used to infer the probability of including r as a hypothetical reaction in a model. Reaction prior $Pr(r)$ for any reaction r in any dataset, is calculated by using the ratio between the frequency of r and the dataset size.

(b) *Co-Occurrence of Reaction Pairs*: For any reaction $r \in R$, if it co-occurs with another reaction in existing pathway maps, then the probabilities of their co-occurring can be used to infer the probability of including r as a candidate reaction. A pair of reactions, (r_1, r) , is said to co-occur if there is a set of one or multiple common compounds, noted as compound set C , among primary products of reaction r_1 and primary substrates of reaction r . The conditional co-occurrence of a pair of reactions (r_1, r) , $Pr(r|r_1)$, is defined as the frequency of reaction pair $r_1 - C - r$, divided by the frequency of reaction r_1 . This co-occurrence indicates the probability of inferring r after seeing r_1 in the dataset.

(c) *Segment Priors*: A pathway segment is a linear sequence of reactions connected by common compounds. Every pathway map is decomposed into a set of pathway segments including from two to six reactions. For any reaction $r \in R$, if it has been seen in pathway segments of existing maps with some prior probability, then these priors can be used to infer the probability of including r as a hypothetical reaction. Segment priors are calculated by dividing the frequency of segments by the size of the dataset.

(d) *Co-Occurrence of Reaction-Segment Pairs*: For any reaction $r \in R$, if it co-occurs with a pathway segment in existing maps, then the probabilities of their co-occurring can be used to infer the probability of including r as a candidate reaction. A reaction-segment pair (r, s) is said to co-occur if the products of reaction r have one or multiple common compounds with the substrates of the first reaction in the segment s , or the opposite, the substrates of reaction r have one or more common compounds with the products of the last reaction in the segment s . The conditional co-occurrence of a reaction-segment pair (r, s) , $Pr(r|r_1)$, is defined as the frequency of segment $r - C - s$ or $s - C - r$, divided by the frequency of segment s . This co-occurrence indicates the probability of inferring r after seeing s in the dataset.

(e) *Gene Co-Occurrence in Complete SEED Organisms*: For any gene g , if it co-occurs with another gene g_1 in known genomes, then the probabilities of their co-occurrence can be used to infer the probability of including g as a candidate gene that may encode for some reaction to fill a network hole. The co-occurrence of gene pairs is calculated by dividing the number of organisms that a pair of genes occurs in by the number of organisms that one gene occurs in. In a count of homology across many organisms, protein families in the SEED—called FIGfams—are used to calculate gene co-occurrence.

(f) *The Co-Occurrence of Gene-Gene Pairs on Gene Clusters in SEED Organisms*: A gene cluster is a group of genes that sit close to each other on chromosomes of an organism. Metabolic genes sitting on the same cluster tend to encode reactions that also construct a pathway segment. The co-occurrence of gene-gene pairs in clusters is calculated by the number of organisms that a gene-gene pair occurs in divided by the number of organisms that the set of genes occurs in.

(g) *Gene Co-Occurrence on Gene Clusters in SEED Organisms*: The co-occurrence of gene pairs in is calculated by the number of organisms that a pair of genes sitting on the same cluster occurs in divided by the number of organisms that one gene occurs in.

3.2 Predictor Construction

Faced with the challenge of searching for hole fillers in a large volume of data, we wish to build computational predictors that infer plausible candidate reactions to reconcile network holes, based on prior knowledge. Specifically, the 23 pieces of evidence extracted above are used, and 23 individual predictors are built according to each piece of evidence. These predictors are categorized according to the seven types of evidence they use.

1. *Predictors Using Reaction Priors*: Five individual predictors, $P_1 - P_5$, are constructed to reflect reaction priors from five different datasets of BiGG models, KEGG reference pathway map, KEGG network modules, KEGG organism pathway maps, and SEED draft models. The underlying idea of these five predictors is that reactions with

higher priors in known datasets are more likely to be selected as candidate reactions than are those with lower priors. Each predictor based on priors from different datasets generates a set of candidate reactions. A scoring function $S(r)$ of a predictor in this group, for any r in one dataset, is set to be $Pr(r)$, which is the prior of reaction r . With scores assigned to all reactions r in the dataset, these reactions are ranked by their scores $S(r)$, and only those reactions with high scores are selected as candidate reactions, noted as R_c .

2. Predictors Using Co-Occurrence of Reaction Pairs: Five predictors, with index of P_6 to P_{10} , are built to include the five pieces of evidence extracted from co-occurrence of reaction pairs in the five datasets. A scoring function $S(r)$ is defined in Equation [1](#) for each of these five predictors. Let us denote all reactions in one of the five datasets as R_d and all reactions in the draft model as R_m . For each reaction pair (r, r_1) , where $r_1 \in R_d, r_1 \notin R_m$ and $r \in R$, the co-occurrence of (r, r_1) , $Pr(r|r_1)$ is calculated. In order to capture all the possible reactions co-occur with reaction r in a local neighborhood of r , the process of calculating $Pr(r|r_1)$ proceeds in five runs. In each run, we consider all reactions r that co-occur with any reaction r_1 that is in the dataset but not in the model. The reaction r with highest co-occurrence with r_1 is inserted into the reaction list R_m and removed from R_d . At the same time, this highest co-occurrence is recorded. Then, with the updated R_m and R_d , the probability of $Pr(r|r_1)$ is recalculated. After five runs, the product of the five highest co-occurrence scores is obtained for any reaction r . The greatest co-occurrence product that contains r is chosen as the score of r , and the pathway of reactions corresponding to this score is considered as a candidate hole filler. All reactions in such a pathway are considered as candidate reactions if this pathway has a high score. This process ensures that reactions in five steps away from r are considered. The number 5 is selected because a pathway segment with at most 6 reactions is used in this work, which in turn is chosen as a tradeoff between the computational cost and the predictive capability of neighbor steps away. After sorting all reactions $r \in R$ by their scores $S(r)$, a list of candidate reactions R_c that have high scores is selected.

$$S(r) = \max \prod_{i=1}^5 \max_{r_1 \in R_d, r_1 \notin R_m} Pr(r|r_1) \quad (1)$$

3. Predictors Using Segment Priors: Five predictors based on segment priors in the five datasets are constructed, called P_{11} to P_{15} . For each reaction $r \in R$, we search for all the segments s in the dataset that contain reaction r , sort these segments s by their priors $Pr(s)$, and assign the maximal probability of these segments to the score of r .

4. Predictors Using Co-Occurrence of Reaction-Segment Pairs: Five predictors, with index from P_{16} to P_{20} , are built according to reaction-segment co-occurrence in the five datasets. Each predictor infers a set of candidate reactions that co-occur and connect, via a common compound set, with some segments in the datasets.

5. Predictors Using Gene Co-Occurrence in Complete SEED Organisms: The gene co-occurrence in all complete genomes of the SEED is applied by building a predictor P_{21} . A scoring function assigns a score $S(r)$ to each reaction $r \in R$ based on the

corresponding gene's co-occurrence with other genes in the model. Like the mechanism of calculating scores for reaction co-occurrence, the genes within five steps from a given gene are considered to cover the local neighborhood.

6. *Predictors Using Co-Occurrence of Gene-Gene Pairs on Gene Clusters in SEED Organisms*: Predictor P_{22} is constructed by using the co-occurrence of gene-gene pairs in the same gene clusters of all SEED organisms. The assumption here is that if one gene g sits in the neighborhood of a collection of genes g_s on the chromosomes in many organisms, then if we see a set of neighbor genes g_s in a new organism, we can assume that g is also present in the new organism.

7. *Predictors Using Gene Co-Occurrence on Gene Clusters in SEED Organisms*: Predictor P_{23} is built based on the gene co-occurrence on the same gene clusters in SEED organisms. The score of r , $S(r)$ can be calculated from the co-occurrence of any gene associated with other genes in the same gene clusters.

3.3 Ensemble of Predictors

Individual predictors based on various evidence may produce inconsistent or incorrect predictions. In order to improve the predictive accuracy and resolve inconsistency in individual predictors, ensemble methods must be incorporated to integrate individual predictors. One simple assembly of individual predictors is to retrieve the results of each predictor and pick the reactions that are predicted by most predictors. This approach assumes there are nonselfish and nonbiased behaviors among all individual predictors.

An alternative approach for integrating individual predictors is to treat the selection of candidate hole-filling reactions as a classification problem. In this approach, two classes, class 0 and class 1, are considered. Any reaction in KEGG ($r \in R$) is assigned to class 1 if it is a hypothetical reaction that should be included in a network, and dedicated to class 0 otherwise. Each predictor generates a score that represents a corresponding attribute; hence 23 individual predictors produce 23 attributes. Every instance is a reaction that includes all 23 attribute values and one extra flag that indicates the class this reaction belongs to. After reducing the hole-filler problem to a classification problem, many classifiers in machine learning can be applied. Four such classifiers—naive Bayes classifier, Bayesian network, multilayer perceptron network, and boosting mechanism—are used in this paper.

4 Experiments and Results

To evaluate the set of computational tools, we designed two groups of experiments. The first group involves a self-consistency check on two reconstructed models of *iJR904* [9] for *Escherichia coli* K-12 and *iSB619* [19] for *Staphylococcus aureus* N315. A set of core metabolic genes selected from [16] is removed from these two models and examined to see how well the predictors would fill network holes caused by the knockout of these core genes. The second set of experiments starts by removing 10% of the reactions in a model at a time, eventually removing 80% of the reactions in the model and recording the change in recovery rate.

4.1 Results of Core Knockouts

We discuss here the results of our experiments with individual predictors and the four cited classifiers.

4.1.1 Results of Individual Predictors and Majority Vote Integrator

Figure 1 shows the true positive (TP) rate and false positive (FP) rate of each individual predictor and the majority vote of individual predictors on the reconstructed *iJR904* model (Figure 1(a)) and *iSB619* model (Figure 1(b)). The x axis shows the predictor indices, with 1 to 23 representing individual predictors and 24 representing the majority vote integrator of all individual predictors. The details of all individual predictors indexed from 1 to 23 are explained in Table 1. The y axis indicates the true positive rate (shown as the first bar, in blue, in each group) or false positive rate (shown as the second bar, in red, in each group) for each predictor or majority vote integrator.

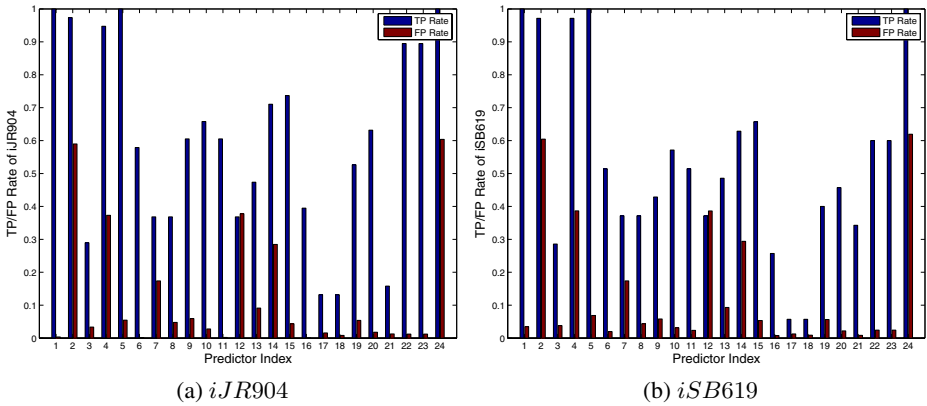


Fig. 1. TP and FP Rates of Individual Predictors and Majority Vote on Reconstructed *iJR904* and *iSB619* Models

From the results of the reconstructed *iJR904* model in Figure 1(a), 12 predictors are considered good because their true positive rates are greater than 0.5 and the false positive rates are smaller than 0.1. However, predictor P_1 , P_6 , P_{11} , and P_{16} should be excluded. The datasets these four predictors use are reaction priors, reaction co-occurrence, segment prior, and reaction-segment co-occurrence in the two reconstructed *iJR904* and *iSB619* models in the BiGG. Hence, these four predictors are largely biased. The remaining eight predictors have good performance in this set of experiments. They are P_5 , P_9 , P_{10} , P_{15} , P_{19} , P_{20} , P_{22} , and P_{23} . Another set of predictors is considered to have fair performance because they have high true positive (with $TP\ rate \geq 0.5$) and relatively high false positive rates (with false positive rates between 0.1 and 0.4) at the same time. This set of predictors comprises P_4 using reaction priors in KEGG organism maps, P_{13} using segment priors in KEGG modules, and P_{14} using segment priors in KEGG organism maps. The majority vote integrator, P_{24} , can recover all knockout reactions if one reaction is said to be recovered by this integrator

if it is recovered by any single predictor. When P_{24} is set to recover a knockout reaction if more than half of individual predictors have voted for that reaction, then the true positive rate is reduced to 0.45, and the false positive rate decreases to a very small ratio of 0.008.

The results of the reconstructed *iSB619* model in Figure 1(b) show a similar pattern for the performance of all predictors. In particular, if the majority vote integrator P_{24} is said to recover a knockout reaction when more than half of individual predictors have voted for that reaction, then the true positive rate is 0.57, and the false positive rate decreases to a very small ratio of 0.001. This results shows that majority vote is good because it can detect approximately 57% of the knockout reactions while keeping the false positive rate very low. We can also conclude that predictors using SEED information, such as predictors P_5 , P_{10} , P_{15} , P_{20} , P_{22} , and P_{23} , perform well. In addition, the evidence from gene clusters, used by P_{22} and P_{23} , strongly suggests candidate hole-fillers.

4.1.2 Classifier Results

The data-mining software Weka [5] is used to train and test four classifiers: naive Bayes classifier, Bayes network, multilayer perceptron network, and AdaBoostM1. Table 2 summarizes the performance of these four classifiers on class 1, which is a class of candidate hole-filling reactions. The classifiers are abbreviated as “NB,” “BN,” “MLP” and “AB,” respectively. The first part is the training error, and the second part is the performance on stratified cross-validation. In each part, three rows represent three different measurements of the performance of classifiers and their accuracy, which is the ratio of correctly classified instances in the test dataset, the true positive rates and false positive rates. The results in Table 2 show that all four classifiers have high accuracy in both training and cross-validation test. Also note that for each of the four classifiers, the cross-validated accuracy is close to that of the training set. We thus conjecture that the classifiers do not overfit the training set [5].

Table 2. Performance of Different Classifiers on Core Knockout Results of *iJR904* Model

		NB	BN	MLP	AB
Training	Accuracy(%)	99.1487	99.236	99.8254	99.9127
	TP	0.863	0.605	0.816	0.921
	FP	0.007	0.004	0	0
Cross Val.	Accuracy(%)	99.0177	99.0613	99.6726	99.8035
	TP	0.816	0.632	0.684	0.868
	FP	0.008	0.006	0.001	0.001

Table 3 shows that all four classifiers have good training results. Each classifier has accuracy greater than 98%, and the TP rate is relatively high and FP rate low. This observation demonstrates that classifiers trained on core knockouts of *iJR904* are capable of recovering core metabolic reactions in the *iSB619* model.

Table 3. Performance of *iSB619* Model for Classifiers Trained on the *iJR904* Model

		NB	BN	MLP	AB
Test	Accuracy(%)	98.7532	99.2261	99.1617	98.9467
	TP	0.343	0.543	0.743	0.914
	FP	0.008	0.004	0.006	0.01

4.2 Results of Random and Subgraph-Based Knockouts

Figure 2 shows the recovery rate of random and subgraph-based knockouts for two predictors on the reconstructed *iJR904* and *iSB619* models. The two predictors are P_5 , which uses reaction priors, and P_{20} , which uses reaction-segment co-occurrence in SEED draft models. In each figure, the x axis is the knockout rate, which is the fraction of the number of knockout reactions and the total number of reactions in the original model. The y axis is the recovery rate of a predictor for corresponding knockout. The black line with circled points represents the change of recovery rate over subgraph-based knockouts. The red line with starred points denotes the change curve of recovery rate on random knockouts.

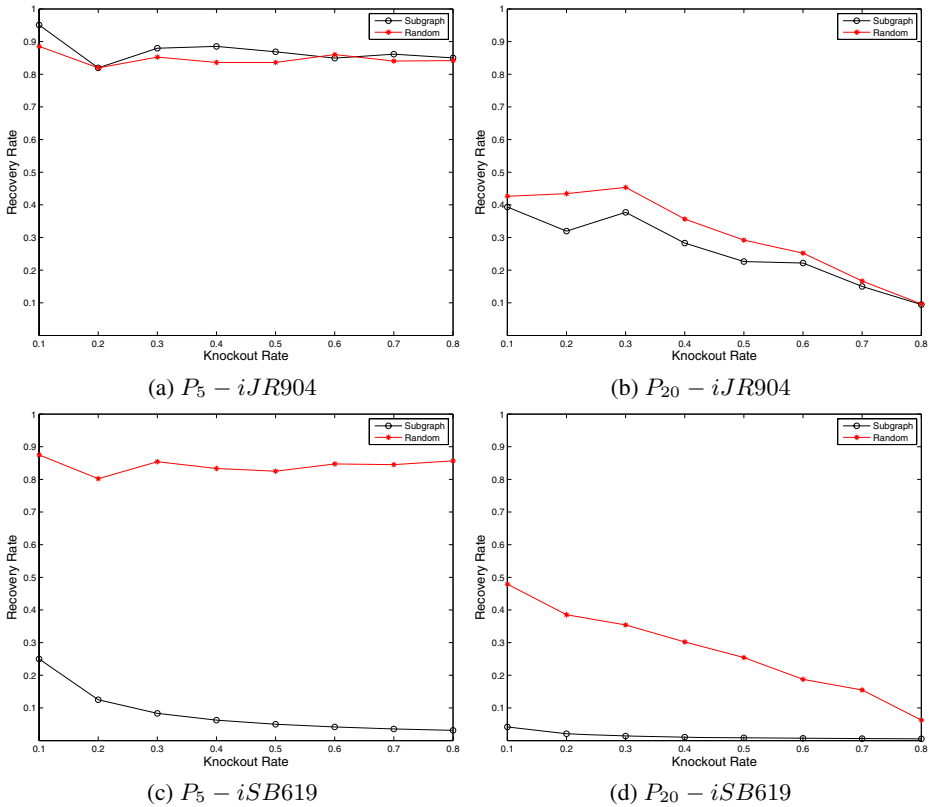


Fig. 2. Recovery Rate of Random and Subgraph-Based Knockouts

In all four subgraphs (a)–(d), we see an overall tendency of declining recovery rate with increased knockout rate. This shows that usually the more reactions that are removed from a model, the fewer of them that can be recovered given the information of remaining reactions in the model. One exception is shown in Figure 2(a), where the recovery rate of predictor P_5 on *iJR904* model stays rather steady as the knockout rate goes from 0.1 to 0.8. The reason is that P_5 is based on the frequency of a reaction in all draft models in SEED, while reactions in these draft models cover the majority of reactions in the *iJR904* model. Therefore, P_5 can recover the majority of the network even though a large proportion of the *iJR904* model is removed. In the cases of *iJR904* model in Figure 2(a) and Figure 2(b), the difference between two lines is trivial. This observation shows that the predictors perform almost equally well for random knockouts and subgraph-based knockouts in the *iJR904* model. In Figure 2(a) and Figure 2(b), however, both predictors recover many more reactions for random knockouts than for subgraph-based knockouts in the *iSB619* model. The reason is that *iSB619* model is not only smaller but also sparser than the *iJR904* model. There are 812 reactions in the reconstructed *iJR904* model, whereas there are 590 reactions in the reconstructed *iSB619* model. Moreover, there are 8,826 pathway segments with length smaller than or equal to 6 in the reconstructed *iSB904* model, whereas the number is 3,054 for the reconstructed *iSB619* model. In summary, Figure 2 shows that the results from our computational predictors agree with the evidence and models.

5 Conclusion

In this paper, we suggest a way to rapidly construct genome-scale models by using a set of reactions to fill network holes. A Bayesian approach is proposed to take into account information gained in databases and to mine through large volumes of data. We present a set of computational tools to extract biological and topological evidence from existing data, construct predictors using different pieces of evidence, and design an ensemble of predictors to integrate and unify individual predictors. By suggesting a collection of candidate hole-fillers computationally, we speed the process of finding hole-filling reactions. We perform a series of experiments in order to evaluate the performance of the approach and computational tools.

The experimental results show that our computational tools recover a large proportion of removed reactions in the reconstructed *iJR904* and *iSB619* models. Moreover, the experiments generate new results that shed light on the properties of metabolic networks and various data and evidence. For example, high true positive rates and low false positive rates for the two predictors using evidence from gene clusters in SEED organisms show that gene clusters are helpful in searching for candidate hole-fillers.

By providing computational tools that support the improvement of draft metabolic models, we expect to generate thousands of genome-scale metabolic models. These tools can be integrated into our efforts of developing a high-throughput scientific workflow [20] to eventually automate the construction of genome-scale metabolic models. The models can be analyzed as a system, and insights can be obtained about properties of organisms, such as genotype-phenotype relationships. With the availability of thousands of biological models, scientists can perform comparative analysis of these models, and a new generation of experimental hypotheses can be achieved.

References

1. Overbeek, R., Begley, T., Butler, R.M., Choudhuri, J.V., Chuang, H.Y., Cohoon, M., de Crécy-Lagard, V., Diaz, N., Disz, T., Edwards, R., Fonstein, M., Frank, E.D., Gerdes, S., Glass, E.M., Goesmann, A., Hanson, A., Iwata-Reuyl, D., Jensen, R., Jamshidi, N., Krause, L., Kubal, M., Larsen, N., Linke, B., McHardy, A.C., Meyer, F., Neuweger, H., Olsen, G., Olson, R., Osterman, A., Portnoy, V., Pusch, G.D., Rodionov, D.A., Rückert, C., Steiner, J., Stevens, R., Thiele, I., Vassieva, O., Ye, Y., Zagnitko, O., Vonstein, V.: IThe Subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. *Nucleic Acids Res.* 33(17), 5691–5702 (2005)
2. Kanehisa, M., Araki, M., Goto, S., Hattori, M., Hirakawa, M., Itoh, M., Katayama, T., Kawashima, S., Okuda, S., Tokimatsu, T., Yamanishi, Y.: KEGG for linking genomes to life and the environment. *Nucleic Acids Res.* 36, 480–484 (2008)
3. BiGG: A Biochemical Genetic and Genomic Database of Large Scale Metabolic Reconstructions, <http://bigg.ucsd.edu/>
4. CellDesigner, <http://www.systems-biology.org/cd/>
5. Weka: Data mining software in Java, <http://www.cs.waikato.ac.nz/~ml/weka/>
6. Feist, A.M., Herrgard, M.J., Thiele, I., Reed, J.L., Palsson, B.O.: Reconstruction of biochemical networks in microbial organisms. *Nat. Rev. Microbiol.* (2008)
7. Palsson, B.: *Systems biology: properties of reconstructed networks*. Cambridge University Press, Cambridge (2006)
8. Reed, J.L., Palsson, B.O.: Minireview thirteen years of building constraint-based in silico models of *Escherichia coli*. *Journal of Bacteriology*, 2692–2699 (2003)
9. Reed, J.L., Vo, T.D., Schilling, C.H., Palsson, B.O.: An expanded genome-scale model of *Escherichia coli* k-12 (ijr904 gsm/gpr). *Genome Biol.* 4(9), R54 (2003)
10. Edwards, J.S., Palsson, B.: Robustness analysis of the *Escherichia coli* metabolic network. *Biotechnology Prog.* 16, 927–939 (2000)
11. Kharchenko, P., Chen, L., Freund, Y., Vitkup, D., Church, G.M.: Identifying metabolic enzymes with multiple types of association evidence. *BMC Bioinformatics* 7(1), 177 (2006)
12. Chen, L., Vitkup, D.: Predicting genes for orphan metabolic activities using phylogenetic profiles. *Geno. Biol.* 7, R17 (2006)
13. DeJongh, M., Formsma, K., Boillot, P., Gould, J., Rycenga, M., Best, A.: Toward the automated generation of genome-scale metabolic networks in the SEED. *BMC Bioinformatics* 8(139) (2007)
14. Green, M.L., Karp, P.D.: A Bayesian method for identifying missing enzymes in predicted metabolic pathway databases. *BMC Bioinformatics* 5(76) (2004)
15. Kharchenko, P., Vitkup, D., Church, G.M.: Filling gaps in a metabolic network using expression information. *Bioinformatics* 20(suppl. 1), I178–I185 (2004)
16. Gil, R., Silva, F.J., Pereto, J., Moya, A.: Determination of the core of a minimal bacterial gene set. *Microbiology and Molecular Biology Reviews* 68(3), 518–537 (2004)
17. Overbeek, R., Begley, T., et al.: The subsystems approach to genome annotation and its use in the Project to Annotate 1000 Genomes. *Nucleic Acids Res.* 33(17), 5691–5702 (2005)
18. Aziz, R.K., Bartels, D., et al.: The RAST server: Rapid Annotations using Subsystems Technology. *BMC Genomics* 9(75) (2008)
19. Becker, S.A., Palsson, B.O.: Genome-scale reconstruction of the metabolic network in *Staphylococcus aureus* n315: an initial draft to the two-dimensional annotation. *BMC Microbiol.* 5(8) (2005)
20. Shi, X., Stevens, R.: SWARM: a scientific workflow for supporting bayesian approaches to improve metabolic models. In: *Proceedings of the 6th international workshop on Challenges of Large Applications in Distributed Environments(CLADE)* (2008)

Parallel Selection of Informative Genes for Classification

Michael Slavik, Xingquan Zhu, Imad Mahgoub, and Muhammad Shoaib*

Department of Computer Science & Engineering, Florida Atlantic University
Boca Raton FL 33431, USA

msslavik@fau.edu, xqzhu@cse.fau.edu, imad@cse.fau.edu

Abstract. In this paper, we argue that existing gene selection methods are not effective for selecting important genes when the number of samples and the data dimensions grow sufficiently large. As a solution, we propose two approaches for parallel gene selections, both are based on the well known *ReliefF* feature selection method. In the first design, denoted by $PReliefF_p$, the input data are split into non-overlapping subsets assigned to cluster nodes. Each node carries out gene selection by using the *ReliefF* method on its own subset, without interaction with other clusters. The final ranking of the genes is generated by gathering the weight vectors from all nodes. In the second design, namely $PReliefF_g$, each node dynamically updates the global weight vectors so the gene selection results in one node can be used to boost the selection of the other nodes. Experimental results from real-world microarray expression data show that $PReliefF_p$ and $PReliefF_g$ achieve a speedup factor nearly equal to the number of nodes. When combined with several popular classification methods, the classifiers built from the genes selected from both methods have the same or even better accuracy than the genes selected from the original ReliefF method.

1 Introduction

Gene expression data are important sources of information for many molecular biology and clinical studies such as genetic diseases profiling [1], identifying potential biomarker signatures for cancers [2], and gene regulatory network reconstruction [4]. Typical microarray experiments output the express values for a large number of genes (*e.g.*, more than 20,000) which impose significant challenges to any tools which intend to interpret the interactions between genes or link the correlations between the genes and diseases. Indeed most machine learning methods are even known to be inaccurate or unstable for high dimensional data, especially when the number of samples is relatively small. Consequently, selecting a number of genes relevant to the tasks at hand has received a lot of attention [3,7,8,9]. Numerous methods have been proposed using computational methods such as Bayes errors [7], Random Forest [8], and Receiver Operating

* This research was carried out when Muhammad Shoaib was doing his postdoctoral research at FAU.

Characteristics (ROC) [9] for gene selection. Research also exists on how to determine the optimal number of genes for a given microarray dataset [6], determining proper sample size for microarray experiments [10], and theoretical analysis of the gene selection from microarray expression data [11].

In order to select important genes, traditional approaches roughly fall into the following two categories: filter and wrapper approaches. In the filter approach, each gene is individually investigated based on its relevance to the targets (*e.g.* diseased or normal tissues), and all genes are ranked based on their relevance values. Typical filtering approaches include *ReliefF* [16], χ^2 [15], Information Gain [5], and many others. The genes selected from the filtering approach are considered important regardless of any classification models. On the other hand, the wrapper approach [18] employs a classification model into the selection process, so that instead of producing a gene ranking list (like the filter approaches do), it produces a gene subset it regards as the best gene set with respect to the current classification model. Typical wrapper selection starts from a single gene (or the whole set of genes), then continuously adds (or drops) genes and investigates the best set which can help build the most accurate models. Because a wrapper approach faces a combinatorial search space, heuristics are usually employed to speed up the search process. Even so, wrapper methods are far more expensive than the filtering methods, and gene subsets selected by them are only relevant for the classifier used in the selection.

For either filter or wrapper gene selection approaches, the selection process is usually time consuming, especially for high dimensional large size datasets. From a high performance computing perspective, one of the cheapest ways to speed up computationally expensive algorithms is parallelizing them to execute on cluster-based supercomputers. Cluster computers are becoming the primary means of supercomputing due to their great and improving cost effectiveness. A cluster supercomputer consists of a number nodes, each of which is typically a stand-alone computer with independent memory, hard disk, and operating systems. The clusters are usually connected in a Local Area Network (LAN) environment based on IP network technology, so all clusters can communicate with each other through high speed switches with Gigabit speeds. Designing algorithms to run in such a distributed fashion can be challenging, given that the program instances are running independently and communicate infrequently and at high cost. An application that can be perfectly distributed will execute N times faster on an N node cluster than on a single machine and is thus said to have linear speedup.

In this paper, we propose two methods, $P\text{Relief}F_g$ and $P\text{Relief}F_p$, to carry out parallel gene selection in an independent and a cooperative manner, respectively. Our results indicate that both methods have close to linear speedup compared to the single machine based $P\text{Relief}F$. The gene selection results from our methods are as good as or even outperform the results of the $P\text{Relief}F$.

The remainder of the paper is structured as follows. Section 2 briefly introduces the *ReliefF* algorithm and its time complexity. The parallel algorithms, $P\text{Relief}F_p$ and $P\text{Relief}F_g$, are introduced in Section 3. In Section 4, these two

algorithms are compared against the single machine based *ReliefF* method in terms of the runtime performance and the quality of the selected genes on a cluster computer. The conclusions and remarks are given in Section 5.

2 Introduction to *ReliefF*

In Figure 1, we list the main procedure of the *ReliefF* algorithm [16]. Given a dataset with m instances $R[i], i = 1, \dots, m$, each of which has a attributes (genes) and a class label (the type of the sample *e.g.* diseased or normal tissue), the output of the *ReliefF* is a ranking list of all genes with the most important genes ranked on the top of the list. The main loop of the algorithm follows 3 steps for each instance $R[i]$

1. find the k nearest instances to $R[i]$ in the same class as $R[i]$
2. find the k nearest instances to $R[i]$ in each other class
3. for each attribute, subtract from the weight the distance (relative to the attribute) between $R[i]$ and its closest neighbors in the same class, and add the distance between $R[i]$ and its closest neighbors in different classes

In order to calculate the difference between two instances with respect to a specific gene A , *ReliefF* defines *diff* as

$$diff(A, I_1, I_2) = \begin{cases} 0 & \text{if } value(A, I_1) = value(A, I_2) \\ 1 & \text{if } value(A, I_1) \neq value(A, I_2) \end{cases} \quad (1)$$

for categorical attributes and

$$diff(A, I_1, I_2) = \frac{|value(A, I_1) - value(A, I_2)|}{max\{A\} - min\{A\}} \quad (2)$$

for numeric attributes. Here $value(A, I)$ is the value of gene A in instance I . R is the vector of instances, each having a class $class(R[i])$. $P(C)$ is the prior probability of class C occurring, and W is the output vector of attribute weights.

The time complexity of the *ReliefF* is $O(m^2a)$ which is quadratic in terms of the number of instances. In this study, we consider gene expression data. A typical example dataset in this class may have $m = 500$, $c = 2$, and $a = 20,000$. Such a dataset has time complexity on the order of 10,000 million, which is computationally expensive for even powerful computers. When gene selection must be performed repeatedly, this can lead to unacceptable delays in calculating results. Thus a need to expedite the algorithm is motivated.

3 Parallel Gene Selection Algorithms

Here we introduce two distributed variants of the *ReliefF*, *PReliefF_p* and *PReliefF_g*, where the theme of the algorithms is to divide the calculation of the *ReliefF* on a cluster computer composed of n nodes.

Algorithm 1: *ReliefF*

```

Input : for each training instance
Output: the vector  $W$  of estimations of the qualities of attributes
1 set all weights  $W[A] \leftarrow 0$ 
2 for  $i \leftarrow 0$  to  $m$  do
3   select the instance  $R[i]$ 
4   find  $k$  nearest hits to  $R[i] \equiv H$ 
5   for each class  $C \neq class(R[i])$  do
6     from class  $C$  find  $k$  nearest misses  $\equiv M(C)$ 
7   end
8   for  $A \leftarrow 1$  to  $a$  do
9      $W[A] \leftarrow W[A] - \sum_{j=1}^k diff(A, R[i], H[j]) / (m \cdot k) +$ 
10     $\sum_{C \neq class(R[i])} \left[ \frac{P(C)}{1 - P(class(R[i]))} \sum_{j=1}^k diff((A, R[i], M(C)[j])) \right] / (m \cdot k)$ 
11  end

```

Fig. 1. *ReliefF* Algorithm

3.1 *PReliefF_p*: Parallel *ReliefF* with Private Weighting

This first variant, *PReliefF_p* (Figure 2), is the most direct subdivision of work across the nodes in the cluster. The set of instances is split in to n subsets, each of approximately equal size. Each node in the cluster processes the instances in one subset, so each node maintains its local weight for all genes based on the set of instances assigned to the current node. At the last stage, all the private weight values are summed a a master node which gives the resulting weight vectors for all genes. This algorithm is called the private weighting version (p) of the algorithm because each node has only private weight values of the elements in the weight vector during execution.

The time complexity analysis of *PReliefF_p* is similar to *ReliefF*. In this case, the main loop is executed m/n times on each node. Each node iterates the main loop in parallel so the main loop requires $O(a \cdot c \cdot m^2/n)$ steps. The final step, summing the partial weight vectors, is $O(n \cdot a)$. Since in most cases $n < m$, we say the time complexity of *PReliefF_p* is

$$O\left(\frac{a \cdot c \cdot m^2}{n} + n \cdot a\right) = O\left(\frac{a \cdot c \cdot m^2}{n} + \frac{n^2 \cdot a}{n}\right) = O\left(\frac{a \cdot c \cdot m^2}{n}\right) \quad (3)$$

3.2 *PReliefF_g*: Parallel *ReliefF* with Global Weighting

PReliefF_g (Figure 3) builds on *PReliefF_p* by introducing a boosting process to reduce the number of genes used to find the distance between two instances. In

Algorithm 2: *PRELiefF_p*

Input : for each training instance
Output: the vector W of estimations of the qualities of attributes

```

1 set all weights  $W[A] \leftarrow 0$ 
2 partition  $R$  into  $n$  non-overlapping subsets  $R_1, R_2, \dots$ 
3 set all partial weights  $W_c[A] \leftarrow 0$  for  $c = 0, 1, \dots, n$ 
4 for each cluster node  $c$  do
5   for  $i \leftarrow 0$  to  $m$  do
6     select the instance  $R[i]$ 
7     find  $k$  nearest hits to  $R[i] \equiv H$ 
8     for each class  $C \neq \text{class}(R[i])$  do
9       | from class  $C$  find  $k$  nearest misses  $\equiv M(C)$ 
10    end
11    for  $A \leftarrow 1$  to  $a$  do
12      |  $W_c[A] \leftarrow W_c[A] - \sum_{j=1}^k \text{diff}(A, R[i], H[j]) / (m \cdot k) +$ 
13      |  $\sum_{C \neq \text{class}(R[i])} \left[ \frac{P(C)}{1 - P(\text{class}(R[i]))} \sum_{j=1}^k \text{diff}((A, R[i], M(C)[j])) \right] / (m \cdot k)$ 
14    end
15  end
16 for  $c = 0$  to  $n$  do
17   |  $W[A] \leftarrow W[A] + W_c[A]$ 
18 end

```

Fig. 2. Parallel gene selection with private weighting

PRELiefF_g, the weight calculations for all genes are carried out in a cooperative manner, where each node updates a global weight vector after processing each instance, and further utilizes the global weight vector to fulfill the weight calculation in the next round. Where *ReliefF* and *PRELiefF_p* use all genes to find nearest neighbors, *PRELiefF_g* progressively excludes the least important genes in terms of weight when it calculates the distance between instances.

The time complexity analysis of *PRELiefF_g* differs from *PRELiefF_p* in two places. First, the main loop of *PRELiefF_g* has an implicit fourth step: selecting the l least important features for exclusion. This step has $O(a \cdot \log(a))$ complexity and is executed once for each of the m/n iterations through the main loop. Second, in *PRELiefF_g*, l (the number of attributes to exclude when computing a distance) ranges from 0 to m during run time. Therefore the operations involved in computing a distance between two instances is reduced on average from $O(a)$ in *PRELiefF_p* to $O(a - m/2)$ in *PRELiefF_g*. Thus the overall time complexity of *PRELiefF_g* is

$$O\left(\frac{m \cdot \left((a - \frac{m}{2}) \cdot c \cdot m + a \cdot \log(a)\right)}{n}\right) \tag{4}$$

The effect this change has on the practical runtime will depend on the relationship between a and m . If $m \ll a$, the first term in the numerator reduces ($(a-m/2) \cdot c \cdot m \approx a \cdot c \cdot m$) and the overall run time will be similar to $PReliefF_p$ as long as $\log a$ is on the same order or smaller than $c \cdot m$. If $m \gg a$, the first term in the numerator reduces to 0 and the overall complexity will become $O(m/n)$, which is significantly faster than $PReliefF_p$. When $m = a$, the time complexity of $PReliefF_g$ is equivalent to $PReliefF_p$, but in the limit it will run twice as fast because the number of computations per distance calculations is cut in half.

The final point to consider is the communications overhead. In the case of $PReliefF_p$, the weight vector is gathered at the root node once, requiring $O(n \cdot a)$ bytes to be transferred. In $PReliefF_g$, the weight vector is sent and received once per iteration of the main loop. The main loop is executed m times total (considering all node), resulting in $O(m \cdot a)$ bytes transferred. Since n is typically small in comparison to m and a , this difference is significant, similar to the difference between linear and quadratic algorithms.

4 Experimental Results

4.1 Data Sets

Our testbed consists of 3 gene expression datasets collected from different resources, including the popular Kent Ridge Biomedical Data Set Repository [19]. The data characteristics of the benchmark datasets are reported in Table 1 (TIS is a sequence dataset for translation initialization site prediction that included mainly for the purposes of demonstrating the algorithm performances on large size high dimensional data). All benchmark datasets contain two types of examples, *i.e.*, positive and negative examples. Using these sets we evaluate the accuracy and execution time performance of the new algorithms, in comparison with the single machine based *ReliefF* method.

Table 1. Data characteristics of the benchmark datasets

Name	# of Samples	# of Genes
AML-ALL Leukemia	72	7,130
Lung Cancer	203	12,601
Translation Initialization Site (TIS)	13375	928

4.2 Cluster Computing

The algorithms are implemented in C on top of the MPICH2 MPI library [17]. This library provides a framework to easily build cluster-based applications. Resulting programs are then executed on an IBM BladeCenter cluster computer consisting of 8 HS20 blade servers, each with Intel Xeon EM64T CPUs operating at 3.8 GHz and 2 GB of RAM. Servers are linked with two 1 Gigabit ethernet controllers. Inter-node communication, handled in large part by the MPICH2 system, is built on an IP network backbone.

Algorithm 3: *PReliefF_g*

Input : for each training instance
Output: the vector W of estimations of the qualities of attributes

- 1 set all weights $W[A] \leftarrow 0$
- 2 partition R into n non-overlapping subsets R_1, R_2, \dots
- 3 set all partial weights $W_c[A] \leftarrow 0$ for $c = 0, 1, \dots, n$
- 4 $l \leftarrow 0$
- 5 **for** each cluster node c **do**
- 6 **for** $i \leftarrow 0$ **to** m **do**
- 7 select the instance $R[i]$
- 8 find k nearest hits to $R[i]$, excluding the l lowest weight attributes $\equiv H$
- 9 **for** each class $C \neq \text{class}(R[i])$ **do**
- 10 from class C find k nearest misses, excluding the l lowest weight attributes $\equiv M(C)$
- 11 **end**
- 12 **for** $A \leftarrow 1$ **to** a **do**
- 13 $W_c[A] \leftarrow W_c[A] - \sum_{j=1}^k \text{diff}(A, R[i], H[j]) / (m \cdot k) +$
 $\sum_{C \neq \text{class}(R[i])} \left[\frac{P(C)}{1 - P(\text{class}(R[i]))} \sum_{j=1}^k \text{diff}((A, R[i], M(C)[j])) \right] / (m \cdot k)$
- 14 **end**
- 15 $l = l + n$
- 16 if $l > 0.9a$, $l = 0.9a$
- 17 **end**
- 18 **end**
- 19 **for** $c = 0$ **to** n **do**
- 20 $W[A] \leftarrow W[A] + W_c[A]$
- 21 **end**

Fig. 3. Parallel gene selection with global weighting

4.3 Runtime Evaluation Results

The runtime performance of the new algorithms is evaluated on 1 to 8 node clusters. Distributed programs are traditionally compared in terms of two metrics, speedup and efficiency, defined below in terms of $T(n)$, the running time on a cluster of n nodes.

$$\text{Speedup}(n) = \frac{T(1)}{T(n)} \tag{5}$$

$$\text{Efficiency}(n) = \frac{\text{Speedup}(n)}{n} = \frac{T(1)}{n \cdot T(n)} \tag{6}$$

Figures 4, 5, and 6 show the performance results. The left axis shows the running time of the algorithm and the right axis shows the speedup, each plotted

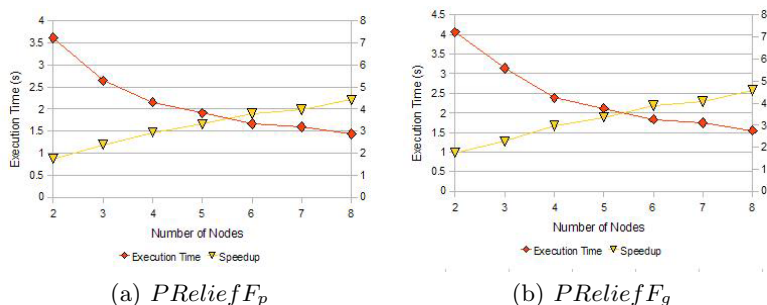


Fig. 4. Run time performance results on the Leukemia Data Set

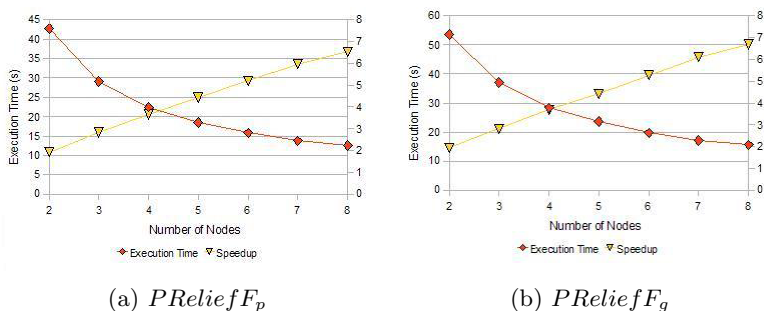


Fig. 5. Run time performance results on the Lung Cancer Data Set

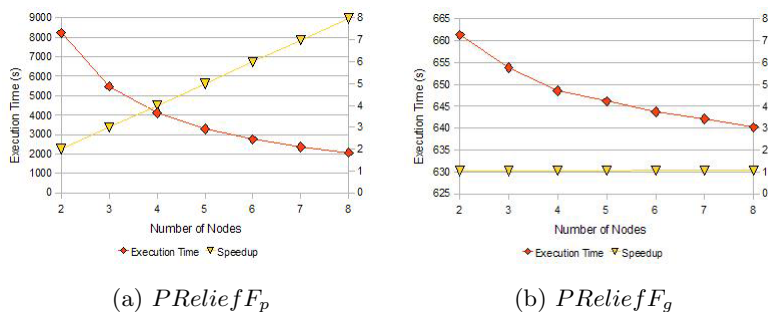


Fig. 6. Run time performance results on the TIS Data Set

vs the number of nodes in the cluster. Perfect speedup is achieved when the speedup equals the number of nodes.

Efficiency gives a measure of how close to perfect speedup an algorithm gets. Under normal circumstances, the maximum efficiency is 100%. Table 2 summarizes the efficiency of the different algorithms.

Table 2. Efficiency of new algorithms

Number of Nodes	$PReliefF_p$			$PReliefF_p$		
	Leukemia	Lung Cancer	TIS		Lung Cancer	TIS
2	87.5	96.2	99.9	87.3	97.3	51.9
3	79.8	94.1	99.9	75.5	93.7	35.0
4	73.4	91.4	99.8	74.4	92.1	26.5
5	66.2	88.6	99.7	67.0	88.3	21.2
6	63.2	86.5	99.7	64.4	87.8	17.8
7	56.7	85.3	99.6	58.1	87.1	15.3
8	55.2	81.6	99.5	57.2	83.6	13.4
Average	68.9	89.1	99.7	69.1	90.0	25.86

In short, the results reported in this section suggest the following.

- $PReliefF_p$ exhibits high efficiency across all data sets, with better efficiency for higher data sets. This relationship is expected, since larger data sets spend a larger proportion of their time in the main loop of the algorithm where the gains from parallelization are made.
- When $a \gg m$ (Leukemia and Lung Cancer), $PReliefF_g$ exhibits high efficiency, and $PReliefF_p$ is marginally faster than $PReliefF_g$. This result is predicted from the time complexity analysis.
- When $m \gg a$ (TIS), $PReliefF_g$ is much faster than $PReliefF_p$. This is also predicted by the complexity analysis.
- The efficiency of $PReliefF_g$ is very low on the TIS data set, implying non-parallelized overhead occupies the majority of the processor time. Investigation shows importing data from disk uses about 6 seconds, and extrapolating the curve in Figure 6(b) shows about 630 seconds of remaining time will remain no matter how many nodes are used. This 630 seconds represents the overhead resulting from data transfer between nodes. The TIS data set has $m = 928$ and $a = 13375$. A gene weight requires 8 bytes and each iteration causes 2 transfers of the weight vector, so a total of about 190 MB is transferred during run time, resulting in an average data transfer rate of about 300 KB/s, which is reasonable given the protocol overhead.

4.4 Accuracy Evaluation Results

For any effort to speedup $ReliefF$ to be useful, it must produce comparable results in terms of accuracy. For this purpose, we measure the accuracies of the classifiers built from the genes selected from proposed methods. More specifically, we filter the data sets to varying numbers of genes using the selection algorithms and build classification models with the resulting data subsets. Two learning methods used in the study include Support Vector Machines (SVM) and k nearest neighbors (k-NN). For all methods, the WEKA tool [14] is used with default parameter settings.

In these experiments, the number of genes selected is varied from 10 to 240. The data set containing the selected genes is used to build the different classifiers

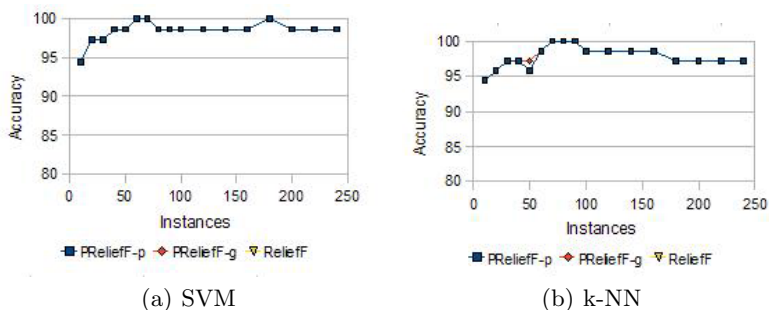


Fig. 7. Leukemia data set accuracy results

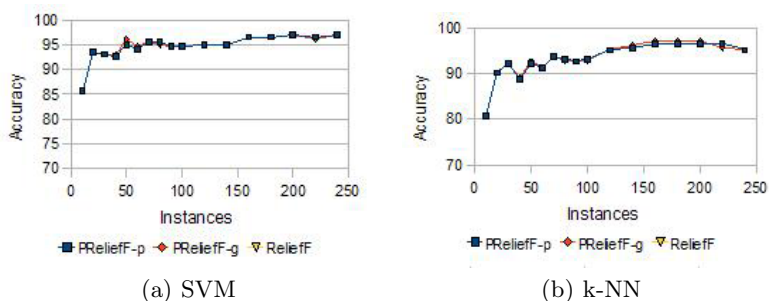


Fig. 8. Lung cancer data set accuracy results

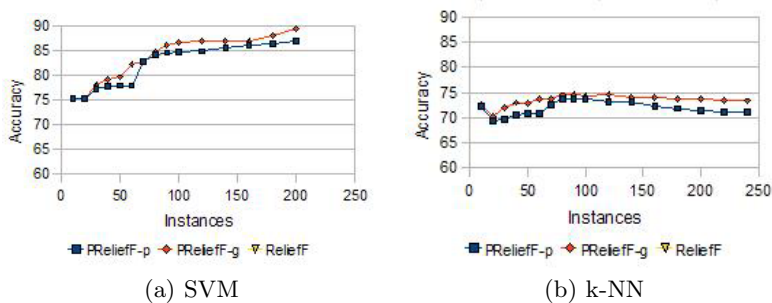


Fig. 9. TIS data set accuracy results

using the above learning methods. Due to the small number of instances in most data sets, 4-fold cross-validation is used to test the model accuracy. Here the accuracy is the percentage of instances that are correctly classified; the true positive rate plus the true negative rate. This metric gives a good indication of the overall accuracy of the method.

Figures 7, 8, and 9 show the accuracy of the classifiers across a range of quantities of genes selected. The results in the Figures 7, 8, and 9 assert that

- $PReliefF_p$ is functionally equivalent to $ReliefF$. This is already known from the algorithm analysis, but is verified here.
- $PReliefF_g$ gives essentially equal accuracy to $ReliefF$ when $a \gg m$ (Leukemia and Lung Cancer). This is predictable since the boosting process has only minimal effect in this case.
- $PReliefF_g$ gives marginally better performance when $m \gg a$ (TIS). By itself this result is not impressive, but recall that $PReliefF_g$ runs much faster on these data sets. In fact, $PReliefF_g$ gives marginally better accuracy in this case while requiring only 4% of the time required by $ReliefF$!

5 Conclusion

Genes expression data are now commonly used in the molecular biology and clinical trial to link the correlations between the expression of certain types of genes and diseases. While many tools exist in finding such correlations, they are generally challenged by the large number of genes under their investigation, especially considering that many genes present themselves in a random manner (or at least the reasons of triggering those genes are yet to be found) or express in all types of tissues (*i.e.* housekeeping genes). Selecting informative genes can eventually help find the genuine interactions between genes and further build enhanced prediction models. Numerous research has shown that selecting a number of informative genes can indeed help build models with better prediction accuracies than the ones trained from the raw data. In this paper, we argued that although many approaches exist for choosing informative genes, these methods, unfortunately, are incapable of handling large datasets, where the algorithms may easily take hours before the users can see the results. Consequently, we proposed two parallel gene selection approaches based on the well known $ReliefF$ feature selection method. Our design employed the master and the worker architecture, and the master dispatches the data to the workers to further carry out the selection process in an independent and a cooperative manner. Experimental results from real-world microarray expression data and an 8 nodes cluster computers show that both versions, $PReliefF_p$ and $PReliefF_g$, linearly speedup with respect to the number of clusters, and the runtime performance of our methods can be as less as 4% of the single machine based method. By using two typical classification methods as learners, we also confirmed that the models trained from the genes selected from our method have the same or even better accuracies than those selected from the original $ReliefF$ method.

References

1. Golub, T., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* 286, 531–537 (1999)
2. Xiong, M., et al.: Biomarker identification by feature wrappers. *Genome Research* 11, 1878–1887 (2001)
3. Baker, S., Kramer, B.: Identifying genes that contribute most to good classification in microarrays. *BMC Bioinformatics* 7, 407 (2006)

4. Segal, E., et al.: Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics* 34(2), 166–176 (2003)
5. Quinlan, J.: *C4.5: Programs for Machine learning*. M. Kaufmann, San Francisco (1993)
6. Hua, J., et al.: Optimal number of features as a function of sample size for various classification rules. *Bioinformatics* 21, 1509–1515 (2005)
7. Zhan, J., Deng, H.: Gene selection for classification of microarray data based on the Bayes error. *BMC Bioinformatics* 8, 370 (2007)
8. Diaz, R., Alvarez, S.: Gene selection and classification of microarray data using random forest. *BMC Bioinformatics* 7, 3 (2006)
9. Mamitsuka, H.: Selecting features in microarray classification using ROC curves. *Pattern Recognition* 39, 2393–2404 (2006)
10. Dobbin, K., et al.: How large a training set is needed to develop a classifier for microarray data. *Clinical Cancer Research* 14(1) (2008)
11. Mukherjee, S., Roberts, S.: A Theoretical Analysis of Gene Selection. In: *Proc. of IEEE Computer Society Bioinformatics Conference*, pp. 131–141 (2004)
12. Li, T., et al.: A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics* 20, 2429–2437 (2004)
13. Statnikov, A., et al.: A comprehensive evaluation of multcategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics* 21(5), 631–643 (2005)
14. Witten, F.E.: *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco (1999)
15. Plackett, R.: Karl Pearson and the Chi-Squared Test. *International Statistical Review* 51(1), 59–72 (1983)
16. Robnik-Šikonja, M., Kononenko, I.: Theoretical and Empirical Analysis of ReliefF and RReliefF *Mach. Learn.* 53, 23–69 (2003)
17. Gropp, W., et al.: *MPICH2 User's Guide* (2008), <http://www.mcs.anl.gov/research/projects/mpich2/index.php>
18. Kohavi, R., John, G.: Wrappers for Feature Subset Selection. *Artificial Intelligence* 97(1-2), 273–324 (1997)
19. Kent Ridge Biomedical Data Set Repository, <http://sdmc.i2r.a-star.edu.sg/rp/>

Simulation Methods in Uncovering New Regulatory Mechanisms in Signaling Pathways

Jarosław Smieja

Silesian University of Technology, Institute of Automatic Control
Akademicka 16, 44-100 Gliwice, Poland
Jaroslaw.Smieja@polsl.pl

Abstract. The paper deals with *in silico* investigation of possible regulatory mechanisms involved in control of signaling pathways. As an example, the Interferon- β activated pathway has been chosen. Though many processes involved in this pathway are known, there are still some that evade explanation. One of them is the nuclear accumulation of the IRF1 protein. Here, three hypotheses are stated and tested numerically, showing that the most likely mechanism involved in this process is an inhibition of IRF1 activation by an yet unrecognized factor.

Keywords: Signaling pathways, Interferon- β , IRF1.

1 Introduction

The term *signaling pathways* (also called regulatory or transduction pathways) relates to the cascade of biochemical processes, initiated either by an external event (e.g., ligand binding to its specific receptor on a cell surface), or by an internal event (e.g., DNA damage). These processes involve creation or degradation of protein complexes, activation of enzymes and usually lead to activation or repression of transcription of genes specific for a given pathway. This results in production of new proteins (or their disappearance, if the genes are suppressed) which may affect earlier stages of the cascade, thus creating positive or negative feedback loops.

Understanding dynamics of signaling pathways involved in immune system responses opens the way for new approaches in drugs development and therapeutics. Following rapid developments in new experimental techniques, mathematical modeling of regulatory pathways that control intracellular biological and chemical processes is gaining increasing interest in the biomedical research. However, our knowledge of the mechanisms regulating intracellular processes is still far from complete. Though mathematical modeling cannot substitute experimental research, through qualitative analysis of existing models it can suggest what regulatory structure is missing (if the model cannot reproduce experimental results) or what is the most promising hypothesis to be tested experimentally.

This paper deals with *in silico* analysis of an unknown mechanisms regulating one of the processes in the Interferon- β (IFN- β) activated signaling pathway. The mathematical model in the form of ordinary differential equations is based on [11]. It

concentrates on one of the phenomena observed in experimental results, which has so far escaped explanation. After brief description of widely recognized structure of the analyzed pathway, three hypotheses are formed in the following sections and the models based on them are tested numerically, showing that only one is acceptable.

2 Canonical form of the IFN- β Activated Pathway

When IFN- β binds to its receptors (IFNAR1/2), the associated tyrosine kinases lead to phosphorylation of STAT1 and STAT2 proteins. Subsequently, phosphorylated STATs form hetero- and homodimers. In cytoplasm, STAT1|STAT2 heterodimers form a complex with an IRF9 protein, called ISGF3. Both STAT1 dimers and ISGF3 complex are transported into the nucleus, where they serve as active transcription factors (TFs), inducing, among others, IRF1 gene transcription. STATs are dephosphorylated by phosphatases both in the nucleus and in cytoplasm. Dephosphorylation results in dissociation of complexes leading to nuclear export of STATs and making them available to subsequent phosphorylation/dephosphorylation cycles. Newly synthesized IRF1 protein translocates to the nucleus, where it subsequently controls late gene expression, including STAT1 gene [2]-[7], [10], [13].

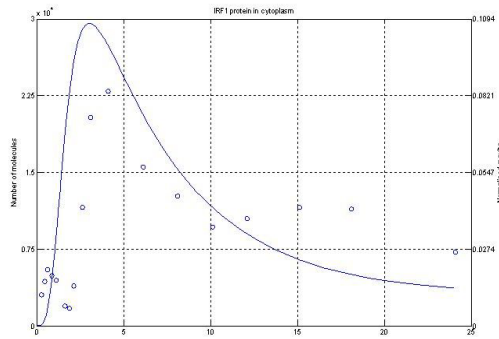


Fig. 1. IRF1 cytoplasmic concentration – simulation for the original model (continuous line) vs. experimental data (circles)

There is a number of mechanisms negatively regulating cell response to IFN, including phosphatases activities, SOCS-based inhibition of STAT phosphorylation, not fully investigated PIASes activities [1], [3], [8], [9], [15], [16].

The original model of Interferon- β stimulated pathway, presented in [11] provided a good fit to experimental data. However, it failed to explain the source of cytoplasmic IRF1 accumulation, observed in the experiments (Fig. 1). According to the laboratory measurements there is a cytoplasmic peak of IRF1 (at approximately, 4 hours of stimulation), reached later than the nuclear peak (after 2.5 hours of stimulation) [11]. These results suggest that there is an additional, unknown process of negative regulation in the pathway.

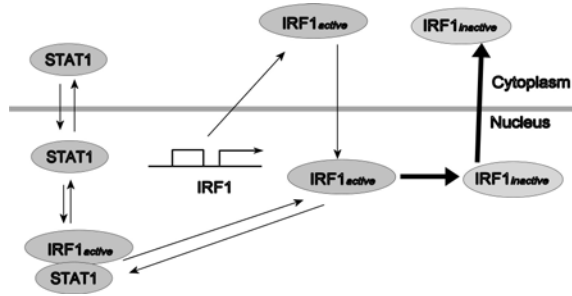


Fig. 2. The part of the original model illustrating processes involving the IRF1 protein. Bold lines indicate hypothesis stated in the original model.

In the original model, the IRF1 protein was assumed to be instantly activated after its production and transported to the nucleus, where it served as a transcription factor. In the nucleus it was assumed to undergo an irreversible inactivation, which resulted in its nuclear export (Fig. 2)

In the following sections, three hypotheses about the regulatory mechanism involved in IRF1 cytoplasmic accumulation will be discussed. All of them require introducing changes to the ODEs describing system dynamics as well as new variables. In order to check how these changes affect simulation results, a whole series of numerical experiments was conducted. In each case, parameters associated with hypothetical processes were changed in a wide range and all possible permutations of parameters were checked.

3 Hypothesis 1 – Creation of IRF1/STAT1 Complexes in the Cytoplasm

The original model took into account creation of STAT1/IRF1 complexes in the nucleus only, assuming that similar process in cytoplasm either does not take place, or can be neglected. Therefore the first hypothesis to be tested assumes that IRF1 can form complexes with unphosphorylated STAT1 in cytoplasm as well (Fig. 3).

Although effects of this process could be negligible initially, if IRF1 nuclear import was very fast, after massive accumulation of STAT 1 in cytoplasm it could play an important role.

Because of STAT1 size the nuclear import of STAT1/IRF1 would have to be facilitated by an importin. Since in such complex the binding site of such importin would be blocked, it is assumed that the complex cannot be transported between the nucleus and cytoplasm.

Similarly as in the original model, the dynamics of IRF1 activation in cytoplasm is neglected and assumed to be almost instant after translation. Therefore, only an active form of IRF1 is present.

As simulation results show, including STAT1/IRF1 complexes in the model does not yield results that match experimental data (Fig.4). Either the peak is reached too fast, or, if initial dynamics is satisfactory, later IRF1 cytoplasmic concentration is

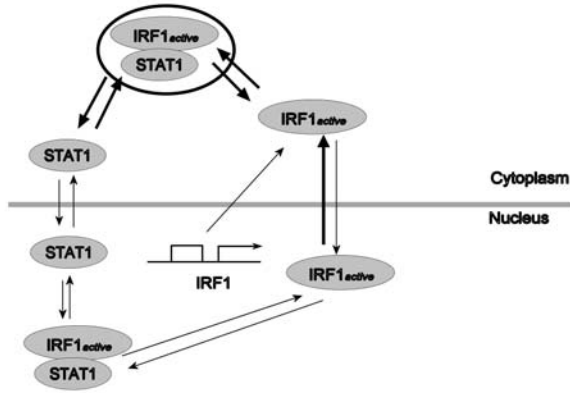


Fig. 3. Part of the original pathway modified as a result of hypothesis I. Bold lines indicate new molecules and processes introduced into the model.

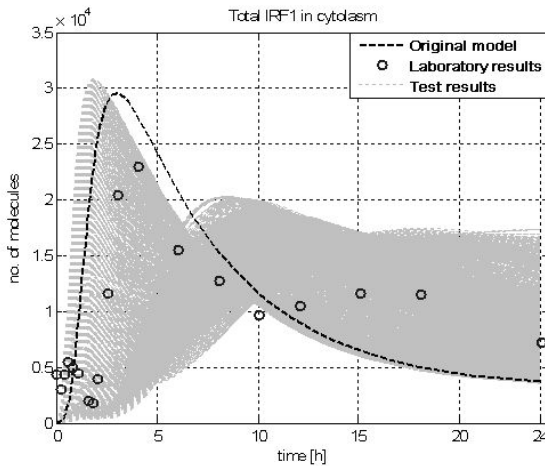


Fig. 4. Cytoplasmic IRF1 dynamics in the model modified by Hypothesis I. Black dashed line, circles and grey lines represent results obtained for the original model, experimental data and simulation results (obtained for various parameters) for the new model.

maintained on the level that is too high. Therefore, one should pursue another explanation of the cytoplasmic IRF1 accumulation.

4 Hypothesis II – Explicit Modeling of IRF1 Activation and IRF1|STAT1 Complexes in the Cytoplasm

Since both in the original model and its modification introduced in the preceding section IRF1 activation has not been explicitly modeled, it might be possible that it is

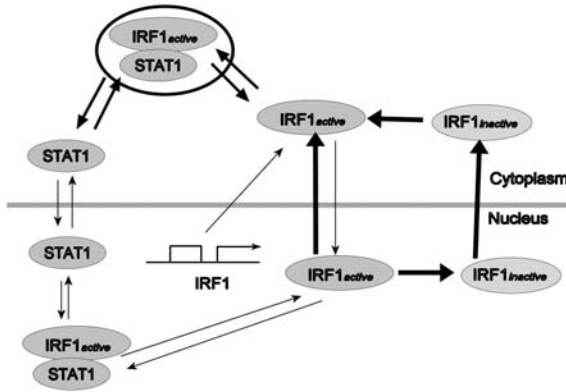


Fig. 5. Part of the original pathway modified as a result of hypothesis II. Bold lines indicate new molecules and processes introduced into the model.

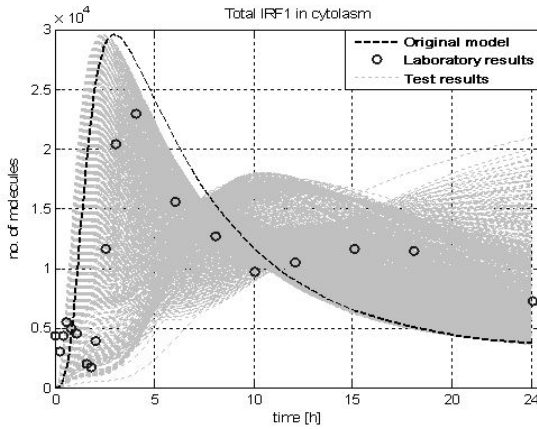


Fig. 6. Cytoplasmic IRF1 dynamics in the model modified by Hypothesis II. Black dashed line, circles and grey lines represent results obtained for the original model, experimental data and simulation results (obtained for various parameters) for the new model.

the dynamics of the activation process that is responsible for the observed phenomenon. In order to check this, another model was tested, in which mRNA translation inactive proteins were produced. They subsequently were activated, following the law of mass action in a first-order process. Additionally, the model allowed for creation of IRF1|STAT1 complexes in the cytoplasm (Fig.5).

An additional advantage of this model is that it allows for inactivation of IRF1 proteins that can be reversible. However, as shown in Fig. 6, for a wide range of parameters it was impossible to reproduce the experimental time profile of IRF1 level.

5 Hypothesis III – Inhibition of IRF1 Activation via an Unknown Repressor X

It is known that IRF1 protein should be activated to act as a TF. This most likely involves phosphorylation by yet unidentified phosphatase [13]. In the previous models it was assumed that the dynamics of such process is very fast comparing to other processes involved in the pathway and therefore can be neglected. Since the changes introduced there have not yielded desired results, another hypothesis is introduced. It will be assumed that the activation of the process is mediated by a kinase that is constitutively present in the cytoplasm. However, this activation can be blocked by another protein that is produced in the pathway, either by binding the kinase or by promoting its degradation. Contrary to the previous models, the dynamics of IRF1 activation is modeled explicitly. Therefore, it is assumed that the translation process yields inactive IRF1 protein (first term in the equation above). Activation of IRF1 protein in cytoplasm is blocked by an unknown X protein and in its absence is a first order process (the last term in the equation). This protein would be coded by a gene whose transcription is induced in the IFN- β activated pathway (Fig. 7). It may be assumed that the TF for this gene should exhibit dynamics similar to other genes activated in this pathway, so it is not necessary to introduce the variable representing concentration of such TF to the model. However, production of the protein should be delayed with respect to IRF1 dynamics. Such delay might result from activation process of X protein. In order to avoid introducing too many unknown molecules into the model it was assumed that the production of X protein is similar to IRF1 protein, and its activation is a second-order time-lag process.

Introduction of an unknown molecule inhibiting IRF1 activation made it possible to obtain cytoplasmic IRF1 accumulation as in the experiments. Moreover, the model proved to be sufficiently robust with respect to parameter changes (Fig. 8).

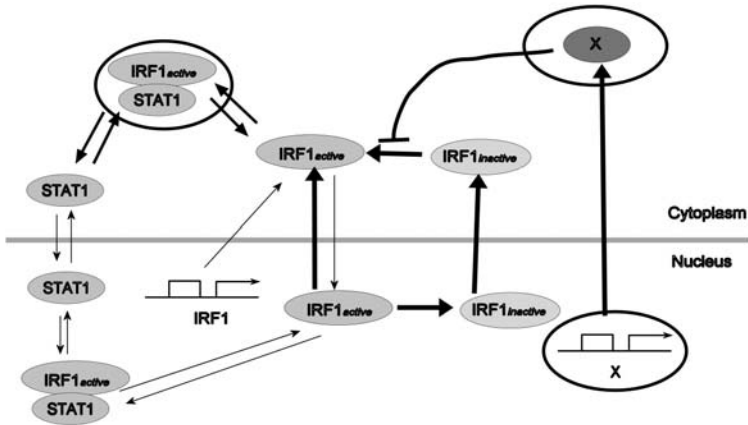


Fig. 7. Part of the original pathway modified as a result of hypothesis III. Bold lines indicate new molecules and processes introduced into the model.

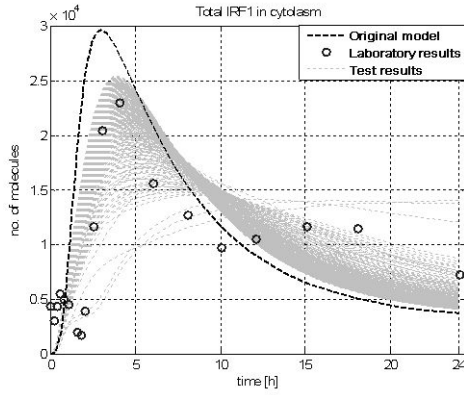


Fig. 8. Cytoplasmic IRF1 dynamics in the model modified by Hypothesis II. Black dashed line, circles and grey lines represent results obtained for the original model, experimental data and simulation results (obtained for various parameters) for the new model.

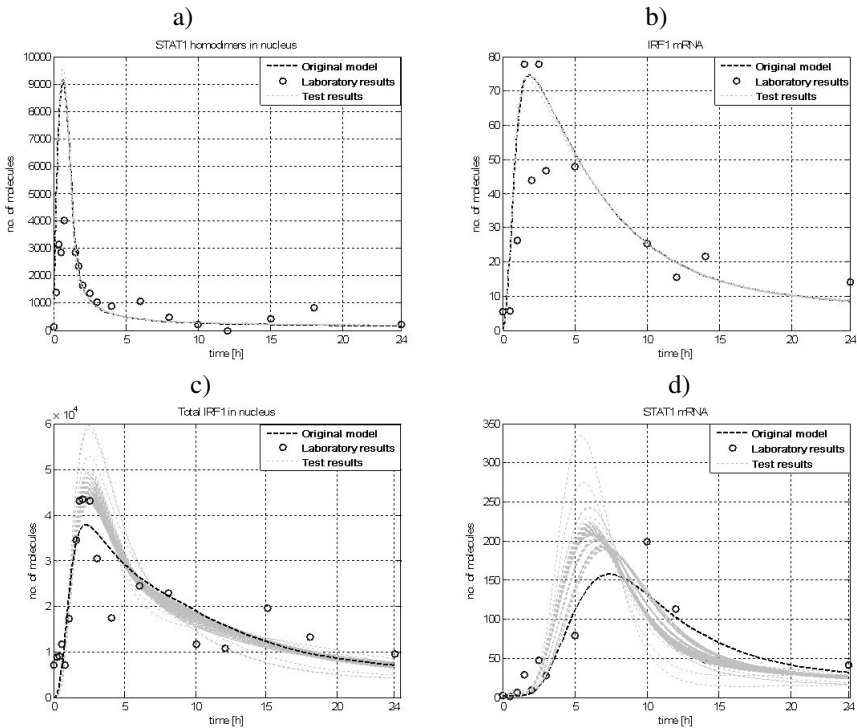


Fig. 9. Original model (black dashed line), experimental data (circles) and simulation results (obtained for various parameters) for the new model (grey lines): a) STAT1 homodimers (TF for IRF1 gene); b) IRF1 mRNA c) nuclear IRF1 protein (TF for STAT1) and d) STAT1 mRNA

Since the changes introduced into the model could influence dynamics of other molecules in the pathway similar plots were produced for them. As illustrated in the Fig. 9, modification introduced in this section has not changed the quality of the fit for them (only TFs and their respective genes are shown). It should be noted that the modifications have virtually no effect on the processes that are upstream of the IRF1 protein production.

6 Conclusions

The paper shows how mathematical modeling can be used to test various hypotheses about structure of regulatory pathways.

Usually the first step in such research is analysis of high-throughput DNA or protein matrices. They, however do not provide enough information to discern causes from effects. Therefore they are usually followed by more focused biochemical experiments. These experiments are very time – and resource-consuming. It is then convenient to support investigation of signaling pathways with mathematical analysis.

Three different models have been tested to find a possible explanation of the cytoplasmic accumulation of IRF1 in the analyzed pathway. Only one of them proved to be able to reproduce the experimental results. While it cannot be considered to be a proof of the hypothesis stated, it shows the direction of the experimental research that is the most promising. Mathematically based approach to analysis of signaling pathways is the most useful, when the hypotheses that are formed include unknown molecules, since they cannot be tested experimentally.

References

- [1] Alexander, W.S., Hilton, D.J.: The role of suppressors of cytokine signaling (SOCS) proteins in regulation of the immune response. *Annu. Rev. Immunol.* 22, 503–529 (2004)
- [2] Bekisz, J., Schmeisser, H., Hernandez, J., Goldman, N.D., Zoon, K.C.: Human Interferons Alpha, Beta and Omega. *Growth Factors* 22(4), 243–251 (2004)
- [3] Chen, W., Daines, M.O., Khurana Hershey, G.K.: Turning off signal transducer and activator of transcription (STAT): The negative regulation of STAT signaling. *J. Allergy Clin. Immunol.* 114, 476–489 (2004)
- [4] Goodbourn, S., Didcock, L., Randall, R.E.: Interferons: cell signalling, immune modulation, antiviral responses and virus countermeasures. *J. Gen. Virol.* 81, 2341–2364 (2000)
- [5] Kalvakolanu, D.V.: Alternate interferon signaling pathways. *Pharmacol Ther.* 100, 1–29 (2003)
- [6] Levy, D.E., Darnell Jr., J.: STATs: transcriptional control and biological impact. *Nat. Rev. Mol. Cell Biol.* 3, 651–662 (2002)
- [7] Pestka, S., Krause, C.D., Walter, M.R.: Interferons, interferon-like cytokines, and their receptors. *Immunol. Rev.* 202, 8–32 (2004)
- [8] Rogers, R.S., Horvath, C.M., Matunis, M.J.: SUMO Modification of STAT1 and Its Role in PIAS-mediated Inhibition of Gene Activation. *J. Biol. Chem.* 278(32), 30091–30097 (2003)
- [9] O’Shea, J.J., Watford, W.: A peek at PIAS. *Nat. Immunol.* 5(9), 875–876 (2004)

- [10] Shuai, K., Liu, B.: Regulation of JAK–STAT signalling in the immune system. *Nat. Rev. Immunol.* 3, 900–911 (2003)
- [11] Smieja, J., Jamaluddin, M., Brasier, A.R., Kimmel, M.: Model-based analysis of Interferon- β induced signaling pathway. *Bioinformatics* 24(20), 2363–2369 (2008)
- [12] Taniguchi, T., Ogasawara, K., Takaoka, A., Tanaka, N.: IRF family of transcription factors as regulators of host defense. *Annu. Rev. Immunol.* 19, 623–655 (2001)
- [13] Taniguchi, T., Takaoka, A.: The interferon- α/β system in antiviral responses: a multimodal machinery of gene regulation by the IRF family of transcription factors. *Curr. Opin. Immunol.* 14, 111–116 (2002)
- [14] Vinkemeier, U.: Getting the message across, STAT! Design principles of a molecular signaling circuit. *J. Cell Biol.* 167(2), 197–201 (2004)
- [15] Wormald, S., Hilton, D.J.: Inhibitors of Cytokine Signal Transduction. *J. Biol. Chem.* 279(2), 821–824 (2004)
- [16] Yasukawa, H., Sasaki, A., Yoshimura, A.: Negative Regulation Of Cytokine Signaling Pathways. *Annu. Rev. Immunol.* 18, 143–164 (2000)

GridSPiM: A Framework for Simple Locality and Containment in the Stochastic π -Calculus

Stephen Tyree, Rayus Kuplicki, Trevor Sarratt, Scott Fujan, and John Hale

Institute of Bioinformatics and Computational Biology, University of Tulsa,
Tulsa, OK 74104, USA

Abstract. Process calculi hold great promise for modeling and analysis of cellular mechanics and behavior. While measured success has been achieved in their simulation of specific biochemical pathways and molecular mechanisms within the cell, several obstacles remain to their widespread adoption and use. Chiefly, these have to do with the difficulty of modeling cell membranes and localized behavior, and limitations on the scalability of the execution model. This paper describes a multi-layered formalism – GridSPiM – that engages notions of concurrency, locality and encapsulation to provide a framework suitable for capturing the key aspects of cellular processes.

1 Introduction

Exploration of the cell has yielded a wealth of knowledge about the molecular mechanisms of many biological interactions and pathways. With this information accumulating rapidly, and with most reactions and mechanisms still largely unexplored, specification and simulation techniques are of great value.

Formal methods in computer science hold the promise of providing a solid theoretical foundation for such techniques. The π -calculus is one such method, capturing concurrency, encapsulation, and structured interactions, each important to understanding biological mechanisms.

Several attempts have been made over the last decade to adapt the π -calculus and other process calculi to biological modeling. Perhaps the most successful of these efforts has been the Stochastic Pi Machine (SPiM) [15]. SPiM's formalism and implementation permit efficient and reasonably scalable stochastic evaluation of π -calculus models.

However, SPiM lacks two capabilities – containment and locality – preventing it from fully bridging the gap between biochemistry and cellular biology. Containment is fundamental to cellular function. Membranes isolate and concentrate reaction machinery and substrates. Proteins embedded in membranes facilitate signaling and actively regulate molecular movement. Locality enforces similar considerations as vast distances within and across cells promote regional concentrations and favor active transport mechanisms over simple diffusion.

In GridSPiM, we place SPiM instances in a framework implementing locality and regulated diffusion. In its simplest form, the framework operates on a hexagonal grid, with each grid space running a SPiM simulation. At fixed intervals of simulation time,

each simulation is stopped and its contents are diffused among its neighboring cells. The user can specify pairwise diffusion rates to restrict movement in a biologically motivated fashion, permitting the modeling of primitive membrane and vesicle structures.

The remainder of this paper details this implementation. Section 2 provides background on biological process calculi and simulation environments. Section 3 gives further consideration to the biological issues motivating this work. Sections 4 and 5, respectively, describe the GridSPiM framework and the results of several simulations. Finally, Section 6 presents future work and concluding remarks.

2 Background

2.1 Process Calculi

Process calculi are a formal mathematical approach to modeling concurrent systems. A process calculus aims to provide a robust syntax, semantics, and set of reductions for the specification of systems in which processes execute and interact in parallel. They have been used to model mobile and networked systems, e.g. simulating the interaction of independent agents on a shared network, and the investigate and prove system properties, e.g. considering cryptographic protocols [1].

Modeled systems in the π -calculus are composed of independent, concurrently-executing processes which interact through named channels. Individual processes are sequential, performing only one operation at a time, but the composition of such processes yields a concurrent system with potentially parallel execution. The process calculus supplies reduction semantics which govern process interaction and system state transitions. The syntax and reduction semantics allow a process calculus to specify a concurrent system formally with well understood and provable properties.

Early examples of process calculi include Communicating Sequential Processes [9], and the Calculus of Communicating Systems [10]. The π -calculus [11][12] was built from the Calculus of Communicating Systems with dynamic systems in mind – collections of processes capable of changing configuration during computation by passing new channel names over existing channels. It has formed the foundation for many subsequent process calculi.

The basic action primitives of the π -calculus include:

$$\begin{aligned} \text{Concurrency:} & \quad P|Q \\ \text{Null Process:} & \quad P|0 \equiv P \\ \text{Replication:} & \quad !P \equiv P|!P \\ \text{Reduction:} & \quad a.P|\bar{a}.Q \rightarrow P|Q \end{aligned}$$

Communication is facilitated by matching named ports ('a' in the below example) and substituting any communicated names ('c' is communicated in place of 'b').

$$\begin{aligned} \text{Substitution:} & \quad \{c/b\}b.d.P \equiv c.d.\{c/b\}P \\ \text{Communication:} & \quad a(b).P|\bar{a}\langle c \rangle.Q \rightarrow \{c/b\}P|Q \end{aligned}$$

Branching is achieved by the choice operator (+) whereby several names can be exposed, each allowing different resulting executions.

Branching: $(a.P + b.Q + c.R) \overline{b}.S \rightarrow Q|S$

This concept of parallel, synchronized, name-passing state machines provides a powerful framework in which to build and evaluate complex models.

2.2 Biological Process Calculi

Process calculi can be useful in formalizing biological problems due to the inherent concurrency of protein and other cellular interactions. A straightforward biological application of the π -calculus was undertaken by Regev [19] to model the RTK-MAPK pathway, a well-studied signal transduction pathway. To accommodate quantitative models, this group extended their work [17] to utilize the stochastic π -calculus [16], a variation of the π -calculus with reaction rates affixed to named communication channels. Reaction rates allow process calculi to reproduce biological timing and quantities more accurately in models.

The addition of primitives for representing and manipulating compartments has further expanded the usefulness of process calculi within biological domains. The Mobile Ambient calculus [5] and its biological derivative, the BioAmbient calculus [18], are π -calculus extensions which provide primitive structures for modeling ambients – self-contained, self-executing compartments which enclose both processes and other compartments. These compartments are useful for modeling cellular boundaries, such as cell membranes. However, due to the approximate nature of ambient compartments, a more biologically-inspired membrane calculus, the Brane calculus [4], was proposed. This process calculus represents compartment boundaries as membranes and introduces membrane operations that more closely simulate the properties of a cellular membrane. The Brane calculus was further extended by the Projective Brane calculus [6], which explicitly enforces the bitonal nature of membranes, therein distilling membrane interactions into even simpler behavior.

2.3 Formal Simulation Environments

Biologically-inspired process calculi have met with measured success in modeling and analysis in the cell biology domain. However, their application has been limited by the sparsity of stochastic execution models suitable for simulation. BioSPi [19] represented an early attempt toward implementing the stochastic π -calculus. BioSPi permitted the exploration of many interesting examples, and, though it suffered from scalability issues, it provided a foundation for further development.

SPiM [15], a descendant of BioSPi, has proven to be the most successful execution model among stochastic process calculi. SPiM provides a well-specified simulation programming language, both command line and graphical interfaces, and a well-developed set of supported operations. SPiM is grounded by a formally defined machine [15] which exploits common structural attributes within simulations to permit an often scalable and efficient execution model.

Some attempts have been made toward formal execution models for Bioambient and Brane calculi [14]. However, lacking well-developed implementations, it is unclear whether these will present the same scalable and intuitive interfaces for simulation as demonstrated by SPiM.

3 Biological Motivation

While process calculi provide a solid general framework for representing biological systems, these systems present several domain-specific issues. We discuss three issues of particular importance for cellular biology – containment, locality, and scalability.

3.1 Biochemical Processes

The seminal work in biological process calculi [19] focused on the cellular protein networks that govern the internal workings of the cell. These systems process external signals, bind and transport molecules, conduct internal reactions, and form cellular structures. Many protein networks are well known, and new knowledge is being gained rapidly. Yet few methods exist for formally specifying known interactions and performing automated analysis and simulation.

Languages and simulation environments derived from process calculi present a useful framework for specifying and simulating these systems. Furthermore, these formalisms may support rigorous composition and analysis [3]. Fitting these lower-level models into realistic cellular environments – with consideration given to locality and membrane containment – can enhance our ability to explore their functions.

3.2 Containment

Membrane interactions are vital to understanding the overall cellular landscape. Membrane boundaries establish the system's hierarchical structure. Beyond merely embodying passive inter-compartmental boundaries, they actively regulate cytoplasmic conditions, transport and filter reaction components, and facilitate intercellular communication and environmental inspection.

These structures must be capable of complexing with other structures and then accurately decomposing into their constituent parts. Additionally, membrane structures must have the ability to change dynamically in response to external and internal stimuli. Hormone interaction, altered metabolism rates, and exposure to varying hydrostatic pressures may affect the composition, and consequently the functional rates of membrane transport and membrane-membrane interaction. Insulin, for example, triggers the activation of glucose transporters on a membrane. The level of activation on a membrane is a function dependent on the concentration of insulin [21]. With complex systems engaging fine-tuned membrane behavior, a greater need for stochastic control of membranes arises.

3.3 Locality

In many process calculi, all interactions occur within the same spatial bounds, so any process can communicate with any other available process. In ambient calculi, membrane boundaries are enforced, but existing within the same location in the ambient hierarchy is sufficient for interaction. However, this unrestricted communication does not accurately reflect the realities within a cell. Vast distances within a cell – at least on the scale of cellular subsystems – limit reactions to components in close proximity to

one another. Since active transport and cellular scaffolding act to close these distances, their impact cannot be ignored [24].

Consider the growth of the *Drosophila Melanogaster* embryo. A gradient of SPZ proteins forms along what eventually becomes the dorsal-ventral axis [22]. These proteins bind with a toll receptor, initiating the endogenous release of a DL protein, which diffuses into the nucleus. Once concentrated in the nucleus, the DL protein acts as a gene repressor, directly determining the differentiation of each cell. The amount of gene repression—based on the concentration of the DL protein—determines the specific fate of a cell, e.g. only certain cells will meet a threshold and develop into the nerve cord. This is largely influenced by the rate of SPZ to toll receptor binding, which in turn, is a function of the gradient of SPZ the cells are exposed to. Without the gradient, cells would be exposed to an equivalent concentration of SPZ, leading to a uniform differentiation of cells.

There are many possible approaches to incorporating notions of locality into a process calculus. These include establishing processes within a coordinate system or a collection of overlapping regions, or using locality ports to encode reaction restrictions. It is unclear how much precision and specification is necessary to effect a realistic system. A stochastic implementation may yield complex emergent behavior from a model far simpler than expected. GridSPiM adopts a simple stochastic grid-based locality framework. Ultimately, empirical feedback will be necessary to determine the sufficiency of this or other approaches.

3.4 Large-Scale Systems

Large scale biological systems exist by the conjunction of reaction pathways organized and regulated by membranes and other structures. A realistic formal model should be able to handle both the lower level details of component interactions and the higher level structures, localities of reaction, and variable environmental conditions present in cellular systems.

Integral to the utility of any formal simulation environment is a scalable execution model capable of evaluating the massive models required for biological exploration. The nature of the modeled interactions suggests a stochastic framework, whereby reductions are selected probabilistically in accordance with model parameters. Parallelization of simulation executions would assist scalability while introducing concerns regarding the accuracy of stochastic aspects as numerous reductions would occur simultaneously potentially skewing relative reaction rates.

4 The GridSPiM Framework

We believe that our framework, GridSPiM, is a viable next step toward achieving the goals outlined in the previous section. This work is centered around the stochastic π -calculus and SPiM, an associated execution model. In this section, we describe this formalism and implementation and several aspects and features of GridSPiM.

4.1 Stochastic π -Calculus

The stochastic π -calculus [16] is a derivative of the π -calculus with reaction rates affixed to channel names. Execution models for the stochastic π -calculus select from a list of available communications between processes based on those reaction rates. Furthermore, delays may be introduced to inhibit reactions for a period of time. Implementations of the Gillespie algorithm [7] are a common method for selecting among the available reductions and delays presented by a system in the stochastic π -calculus. Based on theory of chemical kinetics, the algorithm factors the quantities of reactants and relative reaction rates into the stochastic selection of the next reaction.

4.2 SPiM

SPiM provides a scalable stochastic simulator for reactions that take place in homogeneous solutions. The simulator permits reactions between any pair of processes for which a valid reduction exists. However, this does not address two important problems that should be resolved in a more accurate simulation. First, not all solutions are homogeneous. There may be varying concentrations of entities at different locations in the solution. For example, if there is a reaction manufacturing some product, that product will not instantly diffuse through the solution to make a homogeneous mixture. This may affect reaction rates since a reaction that uses the product cannot take place until that product has made its way to the reaction site. Moreover, SPiM does not address the encapsulation of entities. For example, membranes form spatial boundaries that completely stop many reactions from happening, while gathering and concentrating the necessary components thereby facilitating other reactions.

4.3 Locality-Based Execution Model

Our solution is a simulation built around multiple SPiM instances. Space within the simulation is partitioned across a grid of SPiM simulations, each executing independently. While the adjacency of SPiM instances may be defined within an arbitrary graph, for simplicity we adopt a regular hexagonal grid in this work. Each grid space may have its own process definitions and reaction rates, corresponding to the behavior of local biological processes, e.g. organelle-bound machinery or vesicle-specific conditions. Each SPiM instance is run for a short period of time, after which a set of diffusion rules is applied between runs to allow the diffusion of entities between adjacent spaces. By varying the simulation as well as the diffusion rules across spaces, membranes and organelles can be simulated.

Within GridSPiM, entities correspond to global names for π -calculus processes. While entity names are shared across all SPiM instances in the simulation, each grid space's SPiM instance may have its own process definitions to define the locally-specific behavior of each entity. At the start of a simulation, grid spaces are initialized with a fixed number of each entity. Between iterations, entities are enumerated and are capable of diffusion.

Each space is assigned a type, e.g. cytoplasm, vesicle, or membrane. The type is used to determine what SPiM process definitions and reaction rates modify the entities of that space. The type is also used to control diffusion rates between a space and its neighbors.

4.4 Diffusion

In our implementation, the SPiM instance for each type grid space is run for a fixed amount of simulation time, e.g. 0.1 seconds. After each iteration, diffusion occurs between each adjacent grid space before the instances are run again. Each ordered pair of space types is assigned a rate of diffusion between zero and 1/6 for each entity type. These rates represent the probability that each entity of that type will move from a space of the first type to an adjacent space of the second type. These rates are used to sample a multinomial distribution, and the results of that distribution were used to determine how that entity type from the current space moved. This process is repeated for each entity type in each space and the cumulative results were used to update the grid.

4.5 Static Containment

The GridSPiM framework can support notions of static containment. For example, assume we want to simulate a membrane that allows a protein to pass into the cell, but not out of it. We will define three space types and one entity. The space types are cytoplasm, membrane, and serum. Since this protein can move unhindered in either the cytoplasm or the serum, the rates of diffusion from cytoplasm to cytoplasm and serum to serum are assigned relatively high values. Membrane spaces represent the outside surface of the membrane, so that the diffusion rates from serum to membrane are approximately the same as from serum to serum. To allow for the protein to move into the cell, a non-zero rate of diffusion from membrane to cytoplasm is used. To prohibit the protein from leaving the cell, the rate of diffusion from cytoplasm to membrane is set to zero. This scenario is depicted in Figure 1.

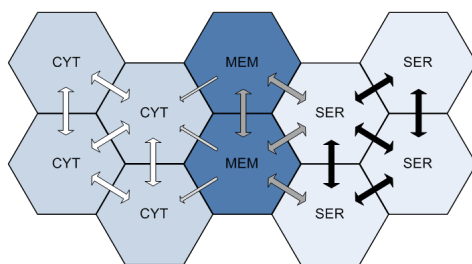


Fig. 1. Static membrane containment in GridSPiM

The static containment structure outlined above can be made more expressive by adding receptors and other transport molecules. To permit only active transport out of the cell, a diffusion scheme could be encoded to operate on the bound structure of a molecule and a transport protein.

The main weakness of this type of containment system is that it does not allow for any changes to the containment structure. The membrane structure can never break open, merge with other membranes, expand or shrink. The rates themselves are also

via endocytosis. Acid pumps change the acidity of the endocytotic vesicle so that the iron is released from the transferrin.

This free iron is able to pass through the vesicle wall and into the cell. Note that the inside of a vesicle is the same surface as the outside of a cell, so that if there were free floating iron in serum, it would be able to pass directly into the cell. The transferrin and receptor molecules can be recycled to the cell's surface, where the transferrin then separates from the receptor and is available for binding with more iron [2].

At any given time, approximately 1/3 of all sites available for binding with transferrin are occupied [13]. Since iron binds tightly with transferrin, there is almost no free iron in serum. Also, the number of receptors present on cells is relatively small, so that the receptors are saturated by transferrin-iron complexes. This means that the uptake of transferrin should depend only on the number of receptors on the cell's surface, up until the point where there is enough iron that all of the transferrin molecules are saturated and free iron is present in serum.

The GridSPiM model for iron absorption defines six different entity types – *iron*, *receptor*, *transferrin*, *transferrin_bound_iron*, *transferrin_bound_receptor*, and *transferrin_bound_both* – and four space types – *cytoplasm*, *vesicle*, *membrane*, and *serum*. The diffusion rates and initial simulation conditions were set so that only iron would exist in cytoplasm, and only iron, transferrin, and iron_bound_transferrin could exist in serum. Every entity could exist on both the membrane and vesicle spaces. The SPiM simulations modeled a reaction to allow iron to bind and unbind to transferrin in serum. This reaction heavily favored binding. On the membrane, iron could bind and unbind with transferrin, and *transferrin_bound_iron* could bind and unbind with receptors. In the vesicle, iron could unbind from *transferrin_bound_both* complex leaving an empty *transferrin_bound_receptor*. Figure 2 uses a graphical notation for the stochastic π -calculus to depict the reactions, and Figure 3 depicts the stochastically favored reactions in each type of grid space.

The rates of the reactions were such that almost all of the receptors existed in the *transferrin_bound_both* state, which means they were almost always bound to a transferrin iron complex. Only *transferrin_bound_both* was allowed to diffuse from the membrane to vesicle spaces. In the vesicle, the same reactions were present as on the membrane, except

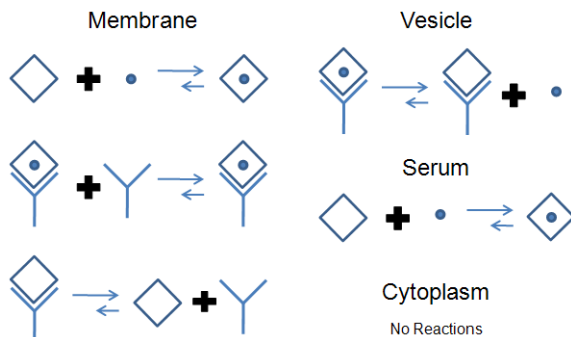


Fig. 3. Stochastically favored reactions by grid space type

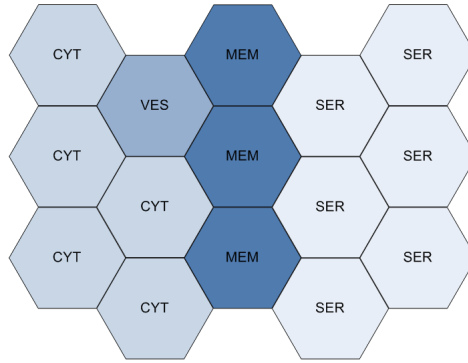


Fig. 4. Layout of grid space types

that the rates were set so that iron would tend to dissociate from transferrin but transferrin and receptors would still exist in the transferrin_bound_receptor state. Iron was given the same rate of diffusion from the vesicle into cytoplasm as it was from the membrane to the cytoplasm, but since there was very little free iron in serum, the iron would tend to enter the cell through a vesicle. The layout of grid space types in the simulation is shown in Figure 4.

Using this experimental model, several different parameters were adjusted to test the results of the simulation against expected values. In each case, the values of interest were the amount of iron that passed into the cell through endocytosis and the amount of iron that passed directly through the membrane. These values were obtained by setting the iron diffusion rate between cytoplasm spaces to zero and monitoring the amount of iron that existed on a space only bordering a vesicle and a space only bordering the membrane.

Experiment 1: Inhibitory Effect of Transferrin. The first experiment was to test transferrin's ability to inhibit varying amounts of iron from entering the cell. For this experiment, the initial amount of iron in serum was varied while keeping the amount of transferrin constant. The amount of initial transferrin in serum was kept constant at 145 molecules per grid space and the amount of iron was given values from 0 to 220. These values represent the relative concentrations of iron and transferrin. Under normal conditions, approximately 1/3 of the binding sites in transferrin are saturated. Each transferrin molecule was modeled with one binding site in our experiments so that normal conditions in the body would be represented with 145 transferrin and 48 iron entities initially in serum. Figures 5 and 6 summarize the results of this experiment.

The results of this experiment were consistent the expected system behavior. The amount of iron that passed into the cell through endocytosis remained constant with varying amounts of iron, as long as two conditions were met. There had to be enough iron that the receptors would always be saturated, and there also had to be more transferrin than there was iron. Since there were relatively few receptors, the first condition was almost always met. The amount of iron that passed into the cell directly through the

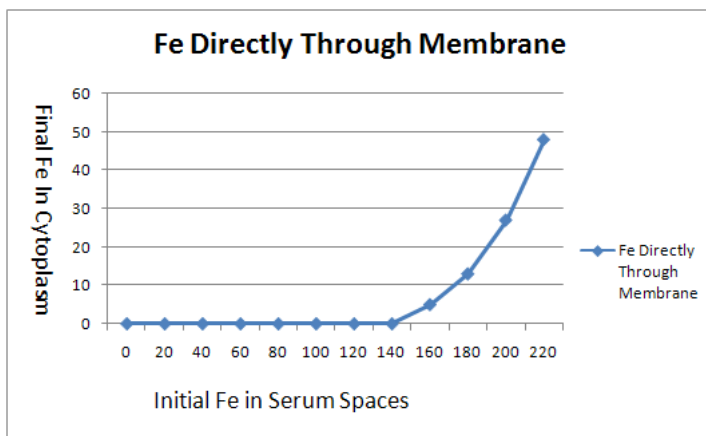


Fig. 5. Iron passed through the membrane compared to initial iron in serum

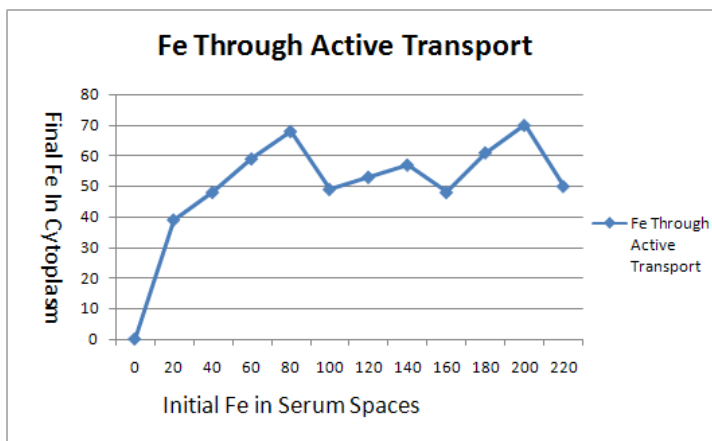


Fig. 6. Iron taken through active transport compared to initial iron in serum

membrane remained effectively zero as long as there were enough transferrin molecules to bind all of the iron. After the initial iron outnumbered the initial transferrin, there was a linear increase in the amount of iron that passed directly through the membrane.

Experiment 2: Effect of Varying Temporal Granularity. The next set of experiments was designed to test the effect that the granularity of the timestep would have on the simulation. The preceding experiment was run for the same total simulation time, but with varying lengths for each iteration and varying numbers of iterations. When the length of the iterations was scaled down, the diffusion rates were scaled down by the same factor. The results of this experiment are summarized in Figure 7.

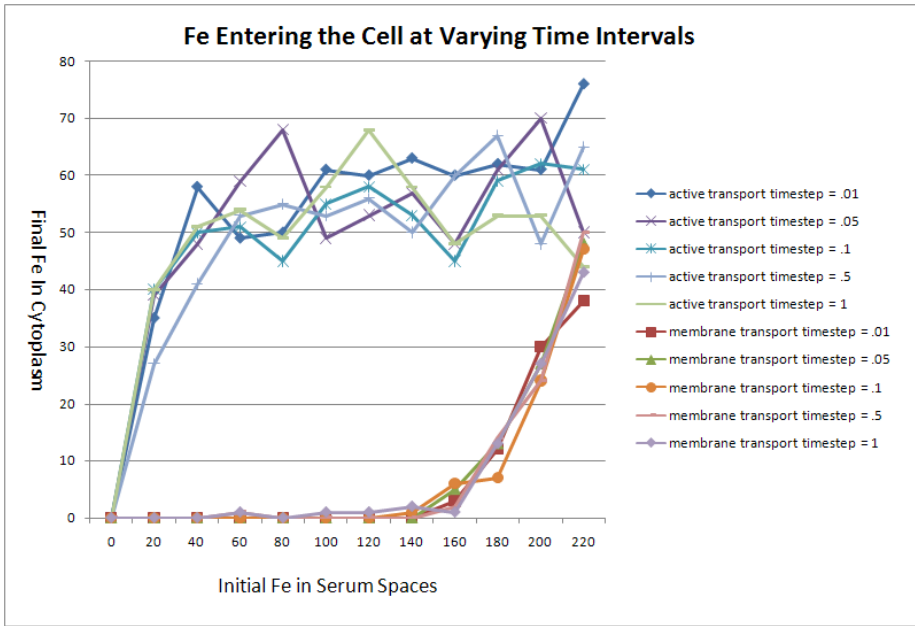


Fig. 7. Effect of timestep granularity

This experiment demonstrated that, in this case, scaling the iteration length and the diffusion rates by the same factor did not significantly alter the results. This implies that time steps may be scaled as needed with only minimal manipulation required to the simulation parameters. Furthermore, it demonstrates that a very fine granularity is not necessarily required to get reasonable results.

5.2 Modeling Fibroblast Growth Factor Pathways

Utilizing knowledge of the endocytic pathway in which Fibroblast growth factor (FGF1) enters a cell, we were able to create a GridSPiM model emulating the behavior of membrane receptors, endocytosis, and routing behaviors for various pathways. FGF has four receptors, FGFR1-4, each with a unique sorting mechanism. Cells expressing FGFR1 have localized FGF in the lysosomes or late endosomes. Conversely, cells treated with FGFR4 exhibit a pathway similar to that of transferrin; a significant portion of the receptors is recycled to the cell surface for reuse [8]. Since these two receptor types offer the greatest contrast in behavior, we only modeled FGFR1 and FGFR4.

In our simulation, bound and unbound receptors were allowed to form vesicles that could move freely about the cell. FGF was released based on stochastic probabilities, and as expected, the released FGF1 formed a gradient when restricted from diffusing – via a diffusion rate of zero specified for FGF1 in the cell. Higher concentrations were found nearer to the membrane, while grid spaces three units in rarely contained FGF1.

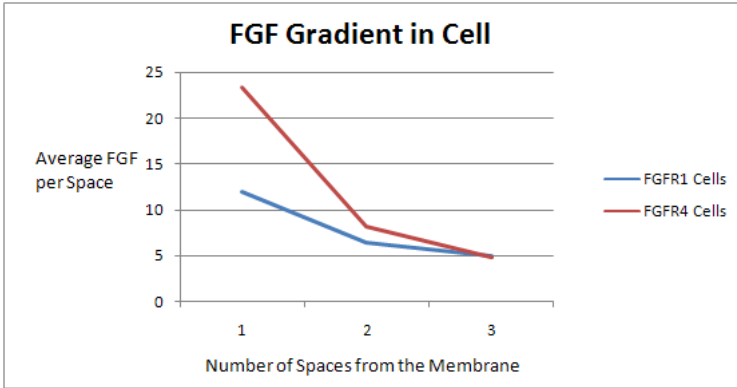


Fig. 8. Transport of vesicles into the cell and release of FGF1 results in an FGF concentration gradient

The differences in pathways, between degradation and recycling, have been attributed to varied targeting of lysines in the receptors by lysosomes [8]. Similarly, our model can reproduce comparable data by only modifying lysosome activity rates. By scaling down the rate of lysis, FGF1 quantities found in the lysosome drops from 80% (0.794 ± 0.038 , $n=10$) of the total FGF found in the cell to 40% (0.395 ± 0.040 , $n=10$). Experimental data for FGFR1 and FGFR4 support these percentages. Cells expressing FGFR1 have a 90% rate of collection for FGF1 in the lysosomes, while cells with FGFR4 have a 45% rate of aggregation in the lysosomes. It appears that a modified scaling mechanism for lysosomal activity would increase the precision of results data; however, without the proper experimental data on lysosomal activity in reference to endosomes containing FGF receptors, scaling rates are derived from convenience. It is noteworthy that our model generated results remarkably similar to a recently developed BioAmbient model of the FGF receptor mechanism in [23].

6 Conclusions and Future Work

Process calculi present syntax and reduction semantics for modeling and analyzing concurrency. Biological applications of the π -calculus and several of its derivatives have yielded valuable tools for specifying and simulating cellular systems. Our work, GridSPiM, has extended previous work with the stochastic π -calculus to provide a framework for explicitly representing locality and containment, two factors critical to the functionality of many biological systems. This yields more expressive models and an execution model capable of simulating systems beyond the scope of previous implementations.

We are currently developing a more expressive formalism for representing and manipulating the containment and locality in our models. The static containment and locality expressions in GridSPiM have proved to be sufficient for several interesting examples, however dynamic membrane interactions and active transport are integral

to cellular function. We intend to maintain the simplicity and efficiency of SPiM at the foundation of our system, with a higher level calculus to manipulate localities and membranes.

References

1. Abadi, M., Gordon, A.D.: A calculus for cryptographic protocols: The spi calculus. In: Fourth ACM Conference on Computer and Communications Security, pp. 36–47. ACM Press, New York (1997)
2. Armstrong, D.: Membrane traffic, from cell to clinic. In: Latchman, D. (ed.) *Basic Molecular and Cell Biology*, 3rd edn., pp. 116–126. BMJ Publishing Group, London (1997)
3. Blossey, R., Cardelli, L., Phillips, A.: Compositionality, stochasticity and cooperativity in dynamic models of gene regulation. *HFSP Journal* 2, 17–28 (2008)
4. Cardelli, L.: Brane calculi. In: Danos, V., Schachter, V. (eds.) *CMSB 2004. LNCS (LNBI)*, vol. 3082, pp. 257–278. Springer, Heidelberg (2005)
5. Cardelli, L., Gordon, A.: Mobile ambients. In: Nivat, M. (ed.) *FOSSACS 1998. LNCS*, vol. 1378, pp. 140–155. Springer, Heidelberg (1998)
6. Danos, V., Pradalier, S.: Projective brane calculus. In: *Computational Methods in Systems Biology*, pp. 134–148 (2004)
7. Gillespie, D.: Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry* 81(25), 2340–2361 (1977)
8. Haugsten, E.M., Sorensen, V., Brech, A., Olsnes, S., Wesche, J.: Different intracellular trafficking of FGF1 endocytosed by the four homologous FGF receptors. *Journal of Cell Science* 118, 3869–3882 (2005)
9. Hoare, C.A.R.: Communicating sequential processes. *Communications of the ACM* 21, 666–677 (1978)
10. Milner, R.: *A Calculus of Communication Systems. LNCS*, vol. 92. Springer, Heidelberg (1980)
11. Milner, R.: A calculus of mobile processes (I and II). *Information and Computation* 100(1), 1–77 (1992)
12. Milner, R.: *Communicating and mobile systems: the π -calculus*. Cambridge University Press, Cambridge (1999)
13. Murtagh, L.J., Whiley, M., Wilson, S., Tran, H., Bassett, M.L.: Unsaturated iron binding capacity and transferrin saturation are equally reliable in detection of HFE hemochromatosis. *American Journal of Gastroenterology* 97, 2093–2099 (2002)
14. Phillips, A.: Efficient, correct abstract machines for stochastic process calculi with mobile compartments. Microsoft Research
15. Phillips, A., Cardelli, L.: A correct abstract machine for the stochastic pi-calculus. In: *Transactions on Computational Systems Biology* (2004)
16. Priami, C.: Stochastic π -calculus. *The Computer Journal* 38(6), 578–589 (1995)
17. Priami, C., Regev, A., Silverman, W., Shapiro, E.: Application of a stochastic name passing calculus to representation and simulation of molecular processes. *Information Processing Letters* 80, 25–31 (2001)
18. Regev, A., Panina, E., Silverman, W., Cardelli, L., Shapiro, E.: BioAmbients: an abstraction for biological compartments. *Theoretical Computer Science, Special Issue on Computational Methods in Systems Biology* 325(1), 141–167 (2004)
19. Regev, A., Silverman, W., Shapiro, E.: Representation and simulation of biochemical processes using the pi-calculus process algebra. In: *Proceedings of the Pacific Symposium of Biocomputing*, vol. 6, pp. 459–470 (2001)

20. Singh, A.K., Coyne, D.W., Shapiro, W., Rizkala, A.R.: Predictors of the response to treatment in anemic hemodialysis patients with high serum ferritin and low transferrin saturation. *Kidney International* 71, 1163–1171 (2007)
21. Spector, A.A., Yorek, M.A.: Membrane lipid composition and cellular function. *Journal of Lipid Research* 26, 1015–1035 (1985)
22. Stathopoulos, A., Levine, M.: Dorsal gradient networks in the drosophila embryo. *Developmental Biology* 246, 57–67 (2002)
23. van Bakel, S., Kahn, I., Vigliotti, M.G., Heath, J.K.: Modelling intracellular fate of FGF receptors with bioambients. *Electronic Notes in Theoretical Computer Science* (2008)
24. Weng, G., Bhalla, U.S., Iyengar, R.: Complexity in biological signaling systems. *Science* 284, 92–96 (1999)

Mutual Information Based Extrinsic Similarity for Microarray Analysis

Duygu Ucar¹, Fatih Altiparmak², Hakan Ferhatosmanoglu¹,
and Srinivasan Parthasarathy¹

¹ Department of Computer Science and Engineering, The Ohio State University,
Columbus, OH

² ASELSAN A.S. Radar, EW, and Intelligence Systems Division, Turkey

Abstract. Genes responding similarly to changing conditions are believed to be functionally related. Identification of such functional relations is crucial for annotation of unknown genes as well as the exploration of the underlying regulatory program. Gene expression profiling experiments provide noisy datasets about how cells respond to different experimental conditions. One way of analyzing these datasets is the identification of gene groups with similar expression patterns. A prevailing technique to find gene pairs with correlated expression profiles is to use linear measures like Pearson's correlation coefficient or Euclidean distance. Similar genes are later compiled into a co-expression network to explore the system-level functionality of genes. However, the noise inherent in microarray datasets reduces the sensitivity of these measures and produces many spurious pairs with no real biological relevance. In this paper, we explore an extrinsic way of calculating similarity of two genes based on their relations with other genes. We show that 'similar' pairs identified by extrinsic measures overlap better with known biological annotations available in the Gene Ontology database. Our results also indicate that extrinsic measures are useful in enhancing the quality of co-expression networks and their functional subnetworks.

1 Introduction and Related Work

Microarray experiments are now being used to profile expression levels of genes under changing experimental conditions. To analyze these profiles in an attempt to answer diverse biological questions, various techniques and ideas have been proposed. Of particular interest to many scientists is the identification of genes whose expression profiles are similar, since genes with similar cellular functions have been theorized to respond similarly to changing conditions [9]. As a result, an efficient similarity measure for microarray analysis is fundamental for understanding the cellular processes [24] and annotating unknown genes.

There has been a growing interest in linking genes whose expression profiles are similar to construct co-expression networks. These networks and their highly modular subnetworks are invaluable sources of information for system-level gene processes [29,4]. Similarity of two genes can be deduced from expression levels

of these genes across all samples [12,29,7]. However, the noise inherent in microarray datasets limits the sensitivity of such analysis. Since any microarray measurement is likely to fluctuate due to many possible sources of error, a similarity based solely on expression measurements of two genes is more error-prone than a similarity based on expression measurements of many genes. In addition, inferring the similarity of two genes based on their relations with a set of other genes will be in accordance with the biological hypothesis about gene products acting as complexes to accomplish certain cellular level tasks [23]. Thus, here we investigate use of extrinsic similarity measures to analyze microarray studies.

The use of extrinsic measures and their advantages have been previously studied for various data mining problems [5,6]. Das et al. [5] proposed using extrinsic measures on market basket data in order to derive similarity between two products from the buying patterns of customers. Palmer et al. [19], defined an extrinsic similarity measure (REP) with an analogy to electric circuits. Both groups concluded that extrinsic measures can give additional insight into the data. Recently, Ravasz et al. [20], took a step towards using extrinsic properties along with the intrinsic similarity. Their measure, the Topological Overlap Measure (TOM), infers similarity of two nodes in a biochemical network in terms of their pairwise similarity as well as the number of their common neighbors. In a previous work we discussed using mutual independence notion to derive an effective extrinsic dissimilarity measures [25].

We introduce application of extrinsic similarity measures for identification of co-expressed genes. We propose extrinsic measures motivated by *Mutual Information* notions from Information Theory. The proposed similarity measures are evaluated on a well-studied cancer microarray dataset [1] obtained with Affymetrix oligonucleotide arrays, as well as a yeast microarray data generated with custom complementary DNA (cDNA) arrays [10]. For both datasets and platforms, we showed that gene pairs obtained by extrinsic similarity measures better overlap with known biological annotations from the Gene Ontology (GO) database when compared to the Pearson's correlation coefficient and the TOM. To further analyze efficacy of extrinsic measures for gene function inference, we constructed co-expression networks by using different measures. We observe that co-expression networks constructed based on extrinsic measures contain less spurious and more biologically verified edges compared to their counterparts generated with other measures. We also studied modular structure of these networks by decomposing them into co-expressed modules. We found that gene modules extracted from Extrinsic Gene Networks are also functionally more homogeneous in comparison.

To summarize, our main contributions in this study are:

- The study of Information Theory concepts, Conditional Mutual Information and Specific Mutual Information, for genes derived from their associations with other genes
- The introduction of extrinsic measures for microarray datasets based on Conditional Mutual Information and Specific Mutual Information

- The demonstration of the efficacy of using extrinsic measures in inferring pairwise gene similarities, constructing co-expression networks, and identifying co-expressed modules.

2 Similarity Measures

To quantify the resemblance of two points, one needs a measure of similarity. Similarity measures can be categorized into two: *extrinsic* and *intrinsic* similarity. An *intrinsic* similarity of two points i and j is purely defined in terms of the values of i and j . On the other hand, an *extrinsic* similarity measure takes into account other points to infer similarity of i and j . Previous studies have shown the usability of extrinsic similarity measures in other domains [5][6]. The standard method to infer similarity of two genes from their expression patterns is to use a linear *intrinsic* similarity such as the Pearson's correlation coefficient. To our knowledge, we are the first to study extrinsic measures for microarray datasets [25].

2.1 Intrinsic Similarity

Intrinsic similarity is purely defined on the points in question. In the context of microarray analysis, the *intrinsic* similarity of two genes is defined on the measured expression levels of these two genes over all samples. In a typical microarray experiment, each gene is expressed at some certain level at each condition which is defined as the expression profile of the gene. More formally, a gene (say, x) is associated with a profile vector (V_x) composed of its expression values over all samples, such that $V_x = [x_1, x_2, \dots, x_n]$, where n denotes the number of samples in the dataset. Thus, *intrinsic* similarity between genes x and y , is a measure defined on their profile vectors, V_x and V_y . A prevailing measure used for inferring similarity of two genes based on their gene profiles is Pearson's correlation coefficient [17]. Throughout our analysis, we employ absolute value of Pearson's correlation scores since both positive and negative correlations can play an important role in gene association. Recently, Ravasz et al [20], proposed the Topological Overlap Measure (TOM) which takes into a step in incorporating external information to infer similarity of two nodes in a biological network. This measure is considered as an improvement over the *intrinsic* similarity which amalgamates an additional external knowledge derived from the network topology (i.e., number of common neighbors).

2.2 Extrinsic Similarity

Extrinsic similarity of two attributes (i.e., genes) is defined over other attributes in the dataset [5]. In general, an extrinsic similarity between two attributes, i and j , can be defined as $ESP(i, j) = \sum_{k \in P} f(i, j, k)$. Here, $f(i, j, k)$ denotes a function that signifies the association between attributes i and j , with respect to a third attribute k . P refers to the set of attributes that will contribute to

the *extrinsic* similarity of attributes i and j . As noted by Das et al [5], proper choice of the attribute set P and function f is crucial for the usefulness of the resulting *extrinsic* measure. Different choices of P and f will result in different similarity notions. Das et. al. [5] preferred to define an extrinsic dissimilarity measure based on the confidence of association rules.

In this work, we propose using Mutual Information of Information Theory to derive efficient extrinsic gene similarity measures. Our final goal is to surmise the similarity of two genes by the similarity of their relation with other genes. We believe that an extrinsic measure for microarray analysis has a twofold advantage over the use of intrinsic measures. First, it may reduce the impact of noise inherent in the dataset on the similarity analysis. It is well known that expression level of each gene is likely to fluctuate due to many sources of variability in a typical microarray analysis. Thus, the similarity deduced from expression levels of two genes is likely to be more error-prone than a similarity deduced from relative positions of these two genes with respect to many other genes. Second, it suits well with the biological hypothesis about genes and gene products acting in the form of complexes (i.e., groups) to accomplish certain tasks in the cell. As hypothesized, two gene products that belong to the same complex behave similarly with the members of this complex. Thus a similarity notion that is defined based on the relation of two genes with other genes can potentially capture the modular structure of the genomic interactions. Moreover, known modular structure of a biological system can be incorporated into the similarity analysis, by defining the P set by using this known structure.

To define proper extrinsic measures, we first need to determine the gene set, P , and the association function, f , that will constitute our measures. For the P set, we make use of the close proximity of each gene determined by an *intrinsic* similarity notion. We propose to use Conditional Mutual Information and Specific Mutual Information as the association functions.

Choice of Attribute Set (P): To derive an efficient *extrinsic* measure, we need an effective gene set that will be used to infer the *extrinsic* similarity of two genes. To compile such a set, we initially identified for each gene a set of genes that are intrinsically similar to that gene. We refer this as the neighborhood list of gene i and define it as $N_i = \{j | j \in G, |r_{ij}| > \kappa\}$, where G denotes the set of all genes in our dataset and $|r_{ij}|$ refers to the absolute value of the Pearson's correlation coefficient between genes i and j . We investigated the effect of the threshold parameter κ in our previous work and observed that size of the neighborhood lists can help us set this parameter [25]. Next, the attribute set P that will be used to infer similarity of two genes is designated as the intersection of their neighborhood lists (i.e., $P = N_i \cap N_j$). Using the common elements in two neighborhood lists, has two important implications. First, it significantly reduces the required number of calculations. Hence, instead of using the whole gene set (G), a smaller size set is taken into consideration for each similarity calculation. Secondly, it filters out irrelevant information which enhances the power of the *extrinsic* measure. Moreover, by using the *intrinsic* similarity to determine elements in set P , we take advantage of both *extrinsic* and *intrinsic*

properties. We believe this will be helpful in reducing the noisy inference that can be introduced into the similarity inference by using each technique separately.

Choice of Association Function (f): Das et al [5], proposed using *confidence* of association rules in an application on market basket dataset. We previously discussed Das’s external dissimilarity measure and its applicability on gene expression datasets [25]. Our analysis showed that it is possible to improve their measure for the task of similar gene identification by using Mutual Independence of genes. We here propose using Conditional Mutual Information and Specific Mutual Information to derive effective extrinsic microarray measures.

To leverage Mutual Information of genes we used probability of occurrence and co-occurrence for genes in the neighborhood lists. Formally we define these probabilities as follows:

Definition 1: Probability of occurrence for a gene i , $P(i)$, is defined as the frequency of encountering that gene in all neighborhood lists. Note that genes with indistinct expression profiles will have higher frequency of occurrence values.

Definition 2: Probability of co-occurrence for two genes, i and j , $P(i, j)$, is defined as the frequency of encountering these two genes together in the neighborhood lists.

Conditional Mutual Information based Gene Similarity: Conditional Mutual Information between variables X and Y , $I(X, Y|C)$, signifies the quantity of information shared between X and Y when C is known. Formally, it is defined as, $I(X, Y|C) = H(X|C) - H(X|Y, C)$ where $H(X)$ signifies the Shannon entropy of the discrete random variable, X . For our calculations, H is defined for the occurrence of a gene in the neighborhood lists. Mutual information calculates the quantity of information shared between X and Y when C is given. $I(X, Y|C)$ is equal to zero iff X and Y are conditionally independent given C . Probabilities of occurrence and co-occurrence are used to calculate Conditional Mutual Information of two genes given neighborhood list of a third gene. A high Conditional Mutual Information between two genes implies that these two genes prefer to co-occur with the same set of genes when a third gene is known to be occurring in the neighborhood lists. If they are not co-occurring with the same set of genes, they will have a smaller Conditional Mutual Information. If two genes bring the same information to the Neighborhood Lists of many third parties, we expect these two genes to be regulated by the same mechanism. Based on this heuristic we define Conditional Mutual Information based Extrinsic Gene Similarity as follows:

$$CMI_P(i, j) = \sum_{k \in P} I(i, j|k = 1) \quad (1)$$

This measure calculates the quantity of information shared by i and j , given that a third gene k is occurring in the neighborhood lists. As can be seen above, the final score is the sum of Conditional Mutual Information between i and j , with respect to all elements in set P . If i and j tend to share the same information, they will have a high CMI similarity value.

Specific Mutual Information based Gene Dissimilarity: The Specific Mutual Information is a measure of association commonly used in the Information Theory to infer mutual dependency. Specific Mutual Information of two variables, X and Y , given their joint distribution, $P(X, Y)$, and individual distributions, $P(X)$ and $P(Y)$, is defined as $\frac{P(X, Y)}{P(X)P(Y)}$, where $P(X, Y)$ is the observed value (O) for joint probability of events X and Y , whereas $P(X)P(Y)$ is its expected value (E). This test can be used to deduce the co-occurrence relation between two genes when their neighbors are considered. If Specific Mutual Information of two genes is 1, it can be concluded that these two genes are independent. In this context, being independent means genes i and j are randomly appearing together in the neighborhood lists. However, if two genes are not independent, occurrence of a gene in a neighborhood list makes it either less probable or more probable for the other gene to occur in that list. Based on this analysis we propose the following *extrinsic* measure to quantify dissimilarity of two genes (i and j).

$$SMI_P(i, j) = \sum_{k \in P} \left| \frac{P(i, k)}{P(i)P(k)} - \frac{P(j, k)}{P(j)P(k)} \right| \quad (2)$$

This definition ensures that two genes having the same co-occurrence relations with their common neighbors are closely related to each other (SMI value close to 0). Whereas two genes that have different independency relations with their common neighbors are dissimilar and associated with higher values of SMI .

We compare the proposed Mutual Information based extrinsic measures with the existing measures in the literature.

3 Domain Based Evaluation

‘Similar’ pairs identified according to different similarity/dissimilarity measures are evaluated based on Pairwise Semantic Similarity measure of Resnik [18]. This measure makes use of known annotations in the Gene Ontology (GO) database. GO is a controlled vocabulary designed to accumulate the result of all investigations in the area of genomic and biomedicine by providing a large database of known associations. Biological relevance of two genes can be quantified with respect to the significance of their shared GO annotations using the Semantic Similarity (SS) measure defined by Resnik [18]. Resnik’s measure is preferred among other semantic similarity measures [11, 13], since it has been shown to outperform the others and suit better to be used for GO analysis [21]. We calculated pairwise semantic similarity for the pairs labeled as similar according to different similarity/dissimilarity measures. We did not take into consideration relations among unannotated genes since there is not enough information to speculate about the biological concordance of these genes.

We then constructed association gene networks by linking the most similar gene pairs identified with respect to alternative similarity definitions. We obtained clusters of densely linked genes from these networks to study their efficacy

in understanding the molecular and biological processes. The obtained clusters are evaluated with an enrichment score that shows the statistical significance of the GO term homogeneity in a cluster. Details of this enrichment score can be found elsewhere [26].

4 Datasets and Pre-processing

For this study, we employ a well-studied cancer dataset and the Rosetta compendium yeast data (i.e., *Saccharomyces cerevisiae*) [10]. Our first dataset is composed of gene expression values of 62 colon tissue samples where the Affymetrix Hum6000 array with 6819 probes is used [1]. 42 of these are collected from colon adenocarcinoma patients and 20 of them are collected from normal colon tissue of the patients. Among all probes, 2000 were selected from 6817 by Alon et al according to the highest minimum intensity [1]. Our second dataset, Rosetta yeast data is obtained using a two-color cDNA microarray hybridization assay [10]. It is composed of 300 compendium experiments on the *Saccharomyces cerevisiae* organism. As suggested by the authors, we used the scale factor for our further analysis, which is defined as the standard deviation of $\log_{10}(\text{ratio})/[\text{error of } \log_{10}(\text{ratio})]$ over all experiments. We perform thresholding, log transformation and normalization (quantile normalization) on these two datasets as suggested by our analysis. In addition to these, we further standardize datasets using a robust standardization method, median absolute deviation (MAD). Genes with zero MAD values implying that they are co-expressed at very similar levels across all of the samples are excluded from further analysis.

5 Experiments

Throughout this section, we discuss usability of extrinsic measures for microarray analysis. First, we present biological relevance of ‘similar’ gene pairs with different measures. We then linked these ‘similar’ genes to construct gene co-expression networks. Each of these networks are partitioned into its functional modules to study the effect of *extrinsic* similarity on the quality of information extracted from these networks.

5.1 Effect on Top ‘Similar’ Pairs

To choose a suitable κ threshold, there are two things that we should take into consideration. First, we want the neighborhood list of a gene to be composed only of genes that are within close proximity of that gene. Second, we do not prefer a set composed of a few genes since this would limit the power of inference based on common neighbors and increase the impact of noise on the final scores. Our previous study showed that average size of the neighborhood lists can guide us while setting the κ parameter [25]. Consequently, we set the κ threshold to 0.5 for the colon cancer dataset and 0.9 for the yeast data, which generates neighborhood lists of size 40 in average.

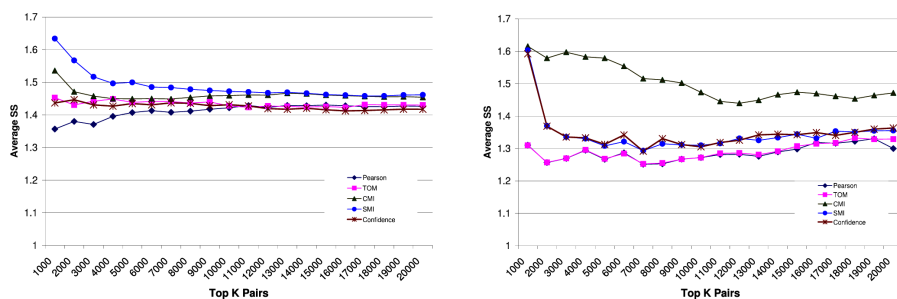


Fig. 1. Average semantic similarity (SS) is calculated for the top ‘similar’ pairs identified via alternative measures from (a) Colon cancer and (b) Yeast microarray datasets. 1K represents the top 1000 pairs identified with each measure.

In our first experiment, we compare gene pairs that are labeled as ‘similar’ according to discussed measures. For each measure, gene pairs are sorted starting from the most ‘similar’ (or least ‘dissimilar’) one. We calculated semantic similarity of all annotated pairs and calculate the average semantic similarity for the whole set of gene pairs. Different number of top scoring pairs (varying between 1000 and 20000) are compared based on their average semantic similarity values. When we analyze the distribution of average semantic similarities, we observe that extrinsic measures outperform existing measures. For both datasets, a significant improvement in semantic similarity is observed.

For the colon cancer dataset, we observe that extrinsic measures significantly overlap with the biological relevance of genes. As can be seen in Figure 1a, the pairs identified with the *SMI* measure show greater biological relevance when compared to the pairs identified by other measures. For the top 1000 pairs, the improvement in the average semantic similarity score is up to 15%, when an extrinsic measure is used instead of an intrinsic one. Since semantic similarity calculations are based on the information content of each GO term which is in the logarithmic scale, this improvement is significant in real world, as our further analysis indicate. Although TOM measure is also able to improve the Pearson’s correlation, this improvement is not as significant as our Mutual Information based extrinsic measures.

When we analyze the yeast dataset, we again observe that extrinsic measures identify biologically more relevant gene pairs. As can be seen in Figure 1b, the improvement is more significant (up to 22%) when top pairs obtained by *CMI* measure are compared to top pairs identified by the standard measure. Note that in contrast to colon cancer dataset, yeast data is obtained using cDNA assays. Our analysis show that extrinsic measures are effective for analysis of both cDNA and oligonucleotide arrays. As can be observed in this figure, TOM contributes even less to standard measure in this case, since mean r values are higher for this dataset.

Our analysis confirm that extrinsic measures better capture the biological relevance of two genes when compared to the standard intrinsic measure. We

believe their power can be attributed to two reasons: the noisy nature of microarray datasets and the functional modularity of genes. *Intrinsic* measures directly possess and reflect the noise inherent in the data since they are purely defined on the expression levels of genes under study. We also believe that since TOM measure is also dependent on the intrinsic measure in its definition, it would also be effected by the noise inherent in these datasets. The poor performance of TOM measure with respect to our extrinsic measures can be attributed to the fact that erroneous measurements will have a more drastic impact on any *intrinsic* or intrinsic based measure. On the other hand, *extrinsic* measures are dependent on more evidence since similarity of two genes are inferred from their relative positions with respect to a set of other genes. Hence, we expect the impact of erroneous measurements to be less severe on the *extrinsic* similarity measures. Our experimental results are also in accordance with this expectation where extrinsic measures produce biologically more relevant pairs. In addition, inferring similarity of two genes from a set of other genes can benefit from the group level interactions known to take place between genes and gene products when accomplishing certain cellular tasks [23].

5.2 Effect on Gene Networks

In this experiment, we constructed gene association networks by linking top similar pairs identified with each measure. Here, nodes represent genes, and two nodes are linked if the corresponding genes are ‘similar’ to each other. To keep the same size for all networks, we only used the top 0.01% of all gene pairs sorted with respect to a similarity/dissimilarity measure. Accordingly, colon cancer networks are composed of 12,438 edges and yeast networks are composed of 74,267 edges. Tightly connected subnetworks of a co-expression network can provide insight into the vital molecular and biochemical processes. Moreover, groups of genes that are densely linked in gene networks have been theorized to have similar cellular functions with great implications for gene annotation at a global scale [9,22,3]. Thus, we extracted and studied densely linked sub-networks of these networks.

To identify densely interacting subnetworks of these networks, we employ a graph partitioning algorithm, Graclus [8], that is shown to be effective in analyzing gene association networks [27]. This algorithm is effective in obtaining balanced-size clusters while minimizing the normalized cuts criterion. To our knowledge, no entirely reliable method exists for identifying correct number of partitions (i.e., k) in a network. That is why, we partitioned colon cancer networks into 100 clusters, and yeast networks into 200 clusters, to make sure reasonable size clusters will be generated at the end. In average 20 genes are located into each partition. Each partitioning is validated using the enrichment score p-value that signifies the homogeneity of each cluster in terms of its known GO annotations. Smaller p-values imply that the grouping is not random and is functionally more homogeneous. A cut-off parameter is used to differentiate significant groups from the insignificant ones. If a cluster is associated with a p-value greater than the cut-off, it is considered insignificant. We used the recommended cut-off of 0.05 for all our validations. The

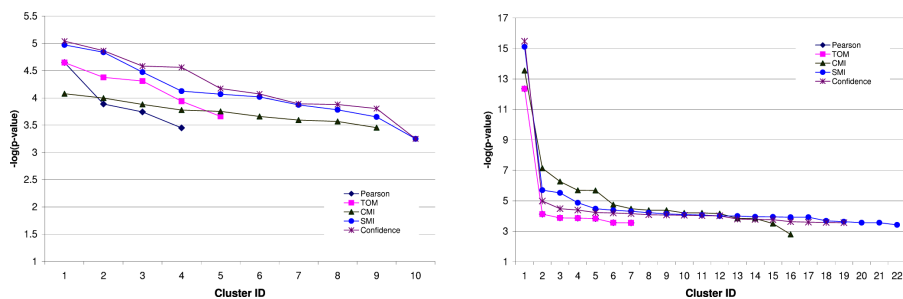


Fig. 2. P-value distribution of significant clusters extracted from (a) Colon Cancer and (b) Yeast gene networks. The y axis represents the $-\log$ of the enrichment score of each corresponding cluster.

p-value distributions for the significant clusters extracted from various gene association networks are shown in Figure 2. As can be observed from the figure, extrinsic similarity measures produce more number of clusters that are significantly enriched with Biological Process GO term annotations. For the colon cancer data, we are able to identify only 4 clusters that are functionally homogeneous when Pearson correlation is used. However, with the use of extrinsic measures this number increases to 10 for SMI and 9 for CMI. Similarly, for the yeast data, number of significant clusters and their significance scores drastically improve when extrinsic measures are used instead of the intrinsic measure. By using SMI measure instead of Pearson's correlation, number of significant clusters that can be deduced from the same data increased more than threefold. These results suggest that using extrinsic measure has a twofold enhancement for co-expression network analysis. First, these measures enhance functional homogeneity of clusters that can be identified with a standard measure as smaller p-values obtained for extrinsic based networks suggest. Also it enables identification of clusters that cannot be detected by standard measures, as evident from the increase in number of significant clusters.

6 Discussion

In this section, we investigate the usability of clusters extracted from different gene similarity networks by running a dataset specific analysis. For this part of our analysis, we analyze the colon cancer dataset which is composed of tumorous and non-tumorous tissues of the human colon and rectum. A more detailed analysis of the significant clusters obtained from the colon cancer data revealed that they can be very useful in understanding and treating the colorectal cancer. We discuss several of these clusters and their relation with colon cancer in the rest of this section.

By using the CMI measure, we obtained a cluster that is annotated with 'aldehyde dehydrogenase (NAD) activity'. Previous studies showed that activity of aldehyde dehydrogenase was measured in primary and metastatic human colonic

¹ Biological Process GO terms are used for this analysis.

adenocarcinomas [14]. We also identified clusters annotated with ‘phospholipase activity’ by employing the CMI measure. It has been shown that Phospholipase D (PLD) has a possible impact on carcinogenesis and its progression [16]. Another cluster obtained with CMI measure is annotated with ‘NF-kappaB binding’. NF-kappaB pathway is shown to be taking part in the regulation of Inhibitors of apoptosis (IAP) family in human colon cancers [28]. Identification of clusters that are known to be related to colon cancer is vital for developing new therapeutic targets and identifying potential tumor markers for colorectal cancer. However, we cannot identify such clusters via standard analysis of the same dataset.

From the *SMI* network, we extracted a cluster that is composed of genes associated with the GO term ‘cytoskeleton-dependent intracellular transport’. Recent evidence indicates that the interaction of a tumor suppressor gene (APC) with the cytoskeleton might contribute to colorectal tumor initiation and progression [15]. That is why, we believe that locating these genes together in a cluster is triggered by the role they play in colon cancer tumorigenesis. Unfortunately, it is still unknown that how APC interacts with the cytoskeleton and how their interaction plays a role in the formation of colorectal tumors [15]. We believe that once functionally coherent clusters are identified, relations between these clusters can be used to reveal function level interactions vital for understanding the cause of some diseases.

7 Conclusion

In this paper, we have introduced the notion of Mutual Information of genes based on their relations with other genes. We have presented two *extrinsic* measures for microarray analysis based on Conditional Mutual Information and Specific Mutual Information. We also discussed a method to employ a previously suggested extrinsic measure for market basket datasets in microarray analysis. We have investigated the efficacy of the proposed measures and run thorough analysis to compare them with standard similarity measures. Our experimental results prove that by using the *extrinsic* measures, it is possible to identify gene pairs that are biologically more relevant. In addition, association networks generated based with these measures are shown to be more informative and useful for further analysis. These results suggest that different similarity notions can reveal different aspects of the same dataset. Previously, we have studied different ensemble techniques to improve clustering results on a scale-free protein interaction network [2]. In the future, we plan to investigate an ensemble approach for integrating different aspects of a dataset captured by different similarity measures.

References

1. Alon, U., et al.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad.* 96, 6745–6750 (1999)
2. Asur, S., Ucar, D., Parthasarathy, S.: An ensemble framework for clustering protein-protein interaction networks. In: *Proc. 15th Annual Int'l Conference on Intelligent Systems for Molecular Biology (ISMB)* (2007)

3. Bader, G., Hogue, C.: An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics* 4(2) (2003)
4. Carter, S., Brechbühler, C., Griffin, M., Bond, A.T.: Gene co-expression network topology provides a framework for molecular characterization of cellular state. *Bioinformatics* 20(14), 2242–2250 (2004)
5. Das, G., Mannila, H., Ronkainen, P.: Similarity of attributes by external probes. In: *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD 1998)*, pp. 23–29 (1998)
6. Das, G., Mannila, H., Ronkainen, P.: Similarity of attributes by external probes. Report C-1997-66, University of Helsinki, Department of Computer Science (October 1997)
7. Datta, S., Datta, S.: Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes. *BMC Bioinformatics* 7(397) (2006)
8. Dhillon, I., Guan, Y., Kulis, B.: Weighted Graph Cuts without Eigenvectors: A Multilevel Approach. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1944–1957 (2007)
9. Eisen, M., Spellman, P., Brown, P., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci.* 95(25), 14863–14868 (1998)
10. Hughes, T., et al.: Functional discovery via a compendium of expression profiles. *Cell*, 102 (2000)
11. Jiang, J., Conrath, D.: Semantic similarity based on corpus statistics and lexical taxonomy. In: *Proc. Int'l Conf. Research in Computational Linguistics, ROCK-LING X* (1997)
12. Lee, H., Hsu, A., Sajdak, J., Qin, J., Pavlidis, P.: Coexpression analysis of human genes across many microarray data sets. *Genome Research* 14, 1085–1094 (2004)
13. Lin, D.: An information-theoretic definition of similarity. In: *Proc. 15th Int'l Conf. Machine Learning* (1998)
14. Marselos, M., Michalopoulos, G.: Changes in the pattern of aldehyde dehydrogenase activity in primary and metastatic adenocarcinomas of the human colon. *Cancer letters* 34(1), 27–37 (1987)
15. Näthke, I.: Cytoskeleton out of the cupboard: colon cancer and cytoskeletal changes induced by loss of apc. *Nature Reviews Cancer* 6, 967–974 (2006)
16. Oshimoto, H., Okamura, S., Yoshida, M., Mori, M.: Increased Activity and Expression of Phospholipase D2 in Human Colorectal Cancer
17. Ostel, B.: *Statistics in research basic concepts and techniques for research workers*. Iowa State University Press, Ames (1963)
18. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, vol. 1, pp. 448–453 (1995)
19. Palmer, C., Faloutsos, C.: Electricity based external similarity of categorical attributes. In: Whang, K.-Y., Jeon, J., Shim, K., Srivastava, J. (eds.) *PAKDD 2003*. LNCS, vol. 2637. Springer, Heidelberg (2003)
20. Ravasz, E., et al.: Hierarchical organization of modularity in metabolic networks. *Science* 297(5586), 1551–1555 (2002)
21. Sevilla, J.L., et al.: Correlation between gene expression and go semantic similarity. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2(4) (2005)
22. Snel, B., Bork, P., Huynen, M.: The identification of functional modules from the genomic association of genes. *Proc. Natl. Acad. Sci.* 99, 5890–5895 (2002)

23. Spirin, V., Mirny, L.A.: Protein complexes and functional modules in molecular networks. *PNAS* 100(21) (2003)
24. Stuart, J., Segal, E., Koller, D., Kim, S.: A gene coexpression network for global discovery of conserved genetic modules. *Science* 302(5643), 249–255 (2003)
25. Ucar, D., Altiparmak, F., Ferhatosmanoglu, H., Parthasarathy, S.: Investigating the use of extrinsic similarity measures for microarray analysis. In: *Proceedings of the BIODDD workshop at the ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (2007)
26. Ucar, D., Asur, S., Catalyurek, U., Parthasarathy, S.: Improving Functional Modularity in Protein-Protein Interactions Graphs Using Hub-Induced Subgraphs. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) *PKDD 2006*. LNCS, vol. 4213, pp. 371–382. Springer, Heidelberg (2006)
27. Ucar, D., Neuhaus, I., Ross-MacDonald, P., Tilford, C., Parthasarathy, S., Siemers, N., Ji, R.: Construction of a reference gene association network from multiple profiling data: application to data analysis. *Bioinformatics* 23(20), 2716 (2007)
28. Wang, Q., Wang, X., Evers, B.: Induction of cIAP-2 in human colon cancer cells through PKC/NF-B. *J. Biol. Chem.* 278, 51091–51099 (2003)
29. Zhang, B., Horvath, S.: A general framework for weighted gene co-expression network analysis. *Statistical Applications in Genetics and Molecular Biology* 4(1) (2005)

Graph Spectral Approach for Identifying Protein Domains

Hari Krishna Yalamanchili and Nita Parekh

Center for Computational Natural Science and Bioinformatics,
International Institute of Information Technology, Gachibowli, Hyderabad, 500032, India
harikrishna@research.iiit.ac.in , nita@iiit.ac.in

Abstract. Here we present a simple method based on graph spectral properties to automatically partition multi-domain proteins into individual domains. The identification of structural domains in proteins is based on the assumption that the interactions between the amino acids are higher within a domain than across the domains. These interactions and the topological details of protein structures can be effectively captured by the protein contact graph, constructed by considering each amino acid as a node with an edge drawn between two nodes if the C_α atoms of the amino acids are within 7\AA . Here we show that Newman's community detection approach in social networks can be used to identify domains in protein structures. We have implemented this approach on protein contact networks and analyze the eigenvectors of the largest eigenvalue of modularity matrix, which is a modified form of the Adjacency matrix, using a quality function called "modularity" to identify optimal divisions of the network into domains. The proposed approach works even when the domains are formed with amino acids not occurring sequentially along the polypeptide chain and no *a priori* information regarding the number of nodes is required.

Keywords: Domain prediction, Protein contact networks, Graph spectral analysis.

1 Introduction

Domains are generally considered as compact, semi-independent units, which form an independent stable folded structure. That is, a domain is able to exist in its 3-dimensional form even when cleaved from the rest of the protein. As a result there are more interactions within a domain than with the rest of the protein [1] and this feature is exploited in most computational approaches. Domains being the basic units of protein folding, function and evolution, their identification and analysis form the first step in understanding the function of proteins. Because it forms a structurally 'separate' region in a three-dimensional protein structure, a structural domain can be determined by two visual characteristics; its compactness and its extent of isolation. Although the boundaries of a domain can be determined by visual inspection, problems occur with domains that are discontinuous or loosely associated. Thus construction of an automated method is not easy.

Decomposition of multi-domain protein structures into individual domains has been traditionally done manually. However, with increasing number of protein structures in the PDB databank [2], it is no more possible to keep the domain databases up to date. There is apparently a need for computationally efficient, reliable and fully automated methods for domain identification. Various approaches have been proposed for domain identification, such as, by detection of hydrophobic core in a protein, e.g., DETECTIVE algorithm [3], exploiting the feature of more connections within a domain compared to those across domains, e.g., the FSSP database [4], DOMAK database [5], or graph theoretic approaches, e.g., Domain Parser [6], Sistla *et al* [7]. Many databases use more than one approach along with manual judgment, for domain identification, e.g., CATH [8], SCOP [9]. A good review of comparison of various algorithmic and expert methods of domain assignment has been recently published by Veretnik *et al* [10].

With large number of protein structures now available in Protein Data Bank (PDB) there exists a need for computationally efficient method for automatically identifying domains in proteins. In this paper, we have proposed a simple and elegant approach for domain identification based on graph theoretic technique, which considers the overall connectivity and topology of the protein structure. It exploits the feature that the interactions between the amino acid residues are higher within a structural domain than across domains. The proposed methodology takes the atomic coordinates of the protein as the input and uses the community detection approach by Newman to identify the most natural subdivisions of the network [11-12]. It requires no prior information about the number of domains and also assesses the quality of the partitions.

2 Materials and Methods

A protein molecule is a chain of amino acids connected by peptide bond and gets folded into a 3D structure involving many interactions, *viz.*, hydrogen bonding, ionic interactions, Van der Waals forces, hydrophobic forces, etc. The main problem for computationally processing such molecules is an efficient data structure which can efficiently capture the structural properties of the molecule, one such data structure is a *graph*. A graph is a collection of points, called *nodes* or *vertices*, with lines connecting pairs of points, called *edges*. Nodes joined by an edge are called *adjacent*, and the degree of a node is defined as the number of its adjacent nodes.

2.1 Constructing Protein Contact Graphs

In this study the protein contact networks have been constructed by considering the backbone C_α atoms as nodes and the edges drawn between two C_α atoms if they are either connected by a peptide bond or are within a cut-off distance, R_c ($\sim 7\text{\AA}$). The value of R_c is chosen as an upper limit to the range of non-covalent interactions that are known to play a significant role in the three dimensional fold of the protein. The construction of the protein network is illustrated in Fig. 1 which shows possible connections of a node i with its neighbors. The Euclidean distance is computed between

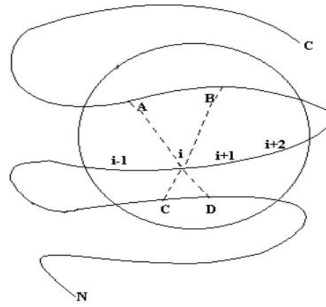


Fig. 1. An edge is drawn from the i^{th} node to all other nodes lying within a circle of radius 7\AA centered at the i^{th} node, i.e., to the nodes $i-1$, $i+1$, $i+2$, A, B, C and D. Nodes A, B, C, D are called spatial neighbors of i (connected by dotted lines) while nodes $i-1$, $i+1$, $i+2$ are called the sequential neighbors of i (connected by continuous lines).

every (i,j) pair, d_{ij} , and an edge is drawn if $d_{ij} < 7\text{\AA}$. The protein contact network can be represented by an $N \times N$ adjacency matrix whose ij^{th} elements are defined as [7]:

$$[A]_{ij} = 1 \text{ if } i \neq j \text{ and } i \text{ and } j \text{ are adjacent vertices (i.e., connected)}$$

$$[A]_{ij} = 0 \text{ if } i = j \text{ or there is no edge between } i \text{ and } j.$$

2.2 Identifying Domains from Protein Contact Graphs

The community detection approach by Newman identifies the most natural subdivisions of the network rather than partitioning a network into pre-defined groups. This is particularly suitable for domain identification problem as one may not have *a priori* knowledge of the number of structural domains in a protein. The basic principle of this approach is to look for divisions of the vertices into two groups so as to minimize the number of edges running between the groups. This is in agreement with the basic feature of domains having large inter-domain connections than between domains. However, a good division of a network into communities is not merely one in which there are few edges between communities but the one with fewer than expected edges between communities. Hence to assess the quality of the division, Newman has defined a measure called *modularity*, which is defined as the difference in the number of edges falling within groups and the expected number in an equivalent network with edges placed at random. For successive decompositions of the protein contact network we use this measure to stop further divisions. The subdivision with the largest value of modularity is considered as the best natural subdivision of the network and is used for further analysis.

Suppose our network contains N vertices. Let the number of edges between vertices i and j be denoted by A_{ij} , which will be either 1 or 0, depending on whether there exists an edge between i and j or not. The quantities A_{ij} are the elements of the adjacency matrix. The expected number of edges between vertices i and j if edges are placed at random is $k_i k_j / 2m$, where k_i and k_j are the degrees of the vertices i and j respectively, and $m = \frac{1}{2} \sum k_i$; is the total number of edges in the network. Thus the modularity, Q , defined as the number of edges falling within groups minus the expected number in an equivalent

network with edges placed at random is given by the sum of $A_{ij} - k_i k_j / 2m$ over all pairs of vertices i, j that fall in the same group. The modularity can be either positive or negative, with positive values indicating the possible presence of community structure. The algorithm works by first constructing the modularity matrix for the network as [12]

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$$

and finding its leading (most positive) eigenvalue and the corresponding eigenvector. The network is divided into two parts according to the sign of the elements of this eigenvector. The process is then repeated for each of the parts, using the generalized modularity matrix,

$$B_{ij}^{(g)} = B_{ij} - \delta_{ij} \sum_{k \in g} B_{ik}$$

where δ_{ij} is the Kronecker δ -symbol, and $B^{(g)}$ is the $n_g \times n_g$ matrix with elements indexed by the labels i, j of vertices within group g . If at any stage a proposed split makes a zero or negative contribution to the total modularity, the corresponding subgraph is left undivided. When the entire network has been decomposed into indivisible subgraphs in this way, i.e., all the elements in a subgraph have their respective principle eigenvector elements of the same sign, the algorithm ends. The split with the largest value of the quality measure, modularity, is considered for further analysis.

When implemented on a multi-domain protein, the above procedure decomposes the protein contact network into a number of structural groups of varying sizes, some of them may even be smaller than a typical domain (< 30 residues). To identify complete compact structural domains, we next implemented a hierarchical agglomerative algorithm by Clauset *et al* [13] on these structural subgroups and join them into a single unit based on the interactions between them. To implement this algorithm, a coarse network graph is constructed, with each structural subgroup as nodes and weighted edges drawn between them based on the number of interactions between the subgroups. That is, if any (x_i, y_j) pair of residues lie within 7\AA distance, where x_i is a node in cluster X and y_j is a node in cluster Y , then the weight, e_{XY} , assigned to the edge between clusters X and Y is given by

$$e_{XY} = \left[\frac{W_{XY}}{(N_X + N_Y)} \right] \times 1000$$

where W_{XY} corresponds to the total number of interactions between the residues of clusters X and Y , normalized by the total size of the two clusters, N_X and N_Y , and 1000 is just a multiplicative constant. Then A_{vw} , an element of the adjacency matrix of this network is defined as:

$$A_{vw} = 1, \text{ if vertices } v \text{ and } w \text{ are connected,}$$

$$0, \text{ otherwise.}$$

with the vertex v corresponding to cluster C_v . Then the fraction of edges that fall within clusters, i.e., that connect vertices that both lie in the same cluster, is

$$\frac{\sum_{vw} A_{vw} \delta(C_v, C_w)}{\sum_{vw} A_{vw}} = \frac{1}{2m} \sum_{vw} A_{vw} \delta(C_v, C_w)$$

where the δ -function, $\delta(i, j) = 1$, if $i = j$ and 0 otherwise, and $m = \frac{1}{2} \sum_{vw} A_{vw}$ is the number of edges in the graph. This quantity will be large for good divisions of the network, in the sense of having many within-community edges, but is not a good measure since it takes its largest value of 1 in the trivial case of all vertices belonging to a single community. However, as before, if we subtract from it the expected value of the same quantity in the case of a randomized network, i.e., $k_v k_w / 2m$, and a useful measure, called modularity is again defined to assess the amalgamation at each step:

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(C_v, C_w)$$

In the agglomerative approach, clusters are repeatedly joined together and the process continued till the amalgamation produces the largest increase in modularity Q . For a network of n vertices, after $n - 1$ such joins the algorithm stops with a single cluster left containing all the nodes [Fig. 2]. Here again, to assess the amalgamation at each step, the modularity measure is computed and amalgamation continued till the grouping with largest modularity Q is obtained. The resultant clusters with largest value Q are then reported as domains. This two-phase approach of domain identification is particularly useful when a protein has domains formed by residues not continuous along the polypeptide chain.

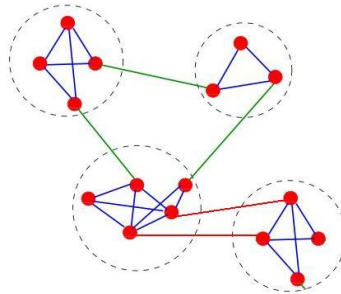


Fig. 2. The solid color filled circles are the initial nodes of the network which are clustered into 4 clusters in the initial phase, represented by dotted circles. Now each dotted circle is considered as a node for the construction of reduced graph and this graph of 4 nodes (clusters) undergoes the aggregation process.

For testing the predictions of the approach discussed here we have compared our results with the annotation of domains provided in CATH database [8] and with the web-based tool DomainParser [6]. A brief description of the methodology used by CATH and DomainParser is given below. CATH uses both computer programs and

human annotations for its domain definition. For each protein, CATH runs three domain decomposition programs, DETECTIVE, PUU and DOMARK [2, 14, 5]. If a consensus is reached among the three programs, CATH adds the consensus result into its database; otherwise it relies on human assignments. DomainParser formulates the domain decomposition problem as a network flow problem and employs the Ford-Fulkerson algorithm to solve it [15]. In this formulation, each residue is represented as a node of a connected network and each residue-residue contact is represented as an edge if the distance between the atoms is within a cut-off value of 4Å. Each interface between two domains is modeled as a minimum flow cut (or bottleneck) of the network. For proteins with multiple domains, the program runs on each of the partitioned substructures recursively until the appropriate stopping criteria are met. The later version of the program combines the structural information of a protein structure including hydrophobic moment profile and neural networks for quality checking of the identified domains.

2.3 Implementation

A Perl script has been written for reading the coordinates of the protein structure from the PDB file to construct the protein contact network by computing the distance matrix. Using this distance matrix, all C_α atoms within 7Å are connected by an edge and this information is stored into an edge list file format wherein each line has two node numbers between which an edge is to be drawn. This edge list file is passed as input to the initial partitioning phase. This edge list file is used to obtain the adjacency matrix and from it the modularity matrix, whose eigenvectors corresponding to the largest eigenvalue are analyzed for graph partitioning. Implementation of partitioning and amalgamation phases are done in C++ using the igraph library [16], which is an open source and distributed under the terms of the GNU GPL for creating and manipulating graphs.

3 Results and Discussion

The analysis has been carried out on a number of multi-domain proteins belonging to different structural classes, *viz.*, α , β , α/β , $\alpha+\beta$, according to the SCOP classification and the results for a few representative proteins are summarized in Table - 7. We discuss in detail below the analysis on two multi-domain proteins 1HLE and 1CDG. In Fig. 3(a) is given the 3-dimensional structure of 1HLE (A), which belongs to the α/β class, and its protein contact network in Fig. 3(b) constructed by drawing an edge between two C_α atoms if they are within a distance 7 Å. Table - 1 summarizes the various steps in the decomposition of the protein network graph at each step of partitioning for the chain A of the protein 1HLE. At each subdivision of the graph, the modularity value is given along with the size and number of clusters formed at each split. The modularity value is seen to increase with every split and has a maximum value for the fourth split after which the modularity value starts falling again. Hence this subdivision is taken as input to the amalgamation phase of the algorithm.

In the amalgamation phase, the clusters corresponding to highest modularity in Table-1 are considered as nodes, *i.e.*, the five clusters, C1, C2, C3, C4 and C5

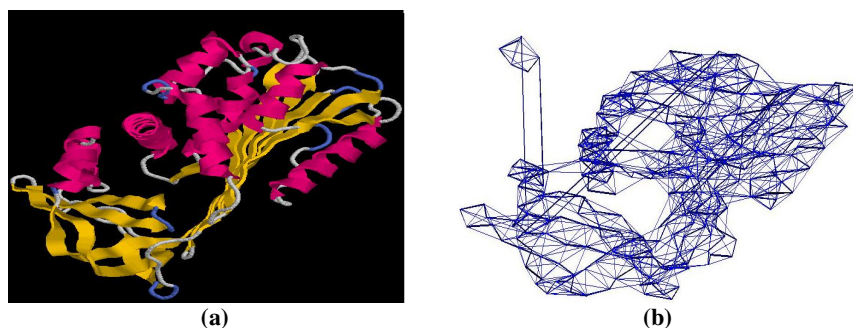


Fig. 3. (a) The 3-dimensional structure of 1HLE(A), and (b) its network graph.

Table 1. Subdivision of protein contact network of 1HLE (A)

Split No.	Modularity	No. of Clusters	Sequence Positions
1	0.412365	2	C1: 1-26,42-76, 98-125, 171-275, 287-314 C2: 27-41, 77-97, 125-170, 276-286, 315-341
2	0.583516	3	C1: 1-26, 171-275 C2: 27-41, 77-97, 116-170, 276-286, 315-341 C3: 42-76, 98-115, 287-314
3	0.634473	4	C1: 1-26, 243-260 C2: 27-41, 77-97, 116-170, 276-286, 315-341 C3: 42-76, 98-115, 287-314 C4: 171-242, 261-275
4	0.638779	5	C1: 1-26 C2: 27-41, 77-97, 116-170, 276-286, 315-341 C3: 42-76, 98-115, 287-314 C4: 171-242, 261-275 C5: 243-260
5	0.634579	6	C1: 1-12 C2: 27-41, 77-97, 116-170, 276-286, 315-341 C3: 42-76, 98-115, 287-314 C4: 171-242, 261-275 C5: 243-260 C6: 13-26

obtained from the fourth split for 1HLE protein, and edges drawn between them based on the number of interactions between the residues in each pair of clusters to construct protein graph. The greedy modularity optimization algorithm by Clauset *et al* is implemented on this protein graph to identify the true structural domains. The output of this amalgamation process is given in Table – 2. It is observed that the third merging step has the highest modularity value, and so the algorithm predicts the most modular decomposition of the structure as two domains, D1 {C1 (1–26), C4 (171-242, 261-275), C5 (243-260) = **1–26 & 171–275**} and D2 {C2(27-41, 77-97, 116-170, 276-286, 315-341), C3 (42-76, 98-115, 287-314) = **27–170 & 276–341**}.

Table 2. Amalgamation of protein cluster graph of protein 1HLE (A)

Merge No.	Modularity	No. of Clusters	Agglomeration of Clusters
1	- 0.227604	4	D1: C2, C3 D2: C1 D3: C4 D4: C5
2	- 0.0221965	3	D1: C1, C5 D2: C2, C3 D3: C4
3	0.0564953	2	D1: C1, C4, C5 D2: C2, C3

The prediction of domains by modularity-based graph spectral approach is compared with the annotation in CATH database and the output of DomainPaser in Table – 3 for 1HLE (A) and a good agreement is observed. It may be noted that domain I is made up of two noncontiguous groups of residues, 1-26 and 171-275 and similarly II, with noncontiguous groups 27-170 and 276-341. Most algorithmic approaches fail to identify such domains formed by groups of residue not contiguous along the polypeptide chain.

Table 3. Comparison of our results with CATH annotation and DomainParser output.

PDB ID	Class	CATH Results	DomainParser Results	Spectral Graph Method
1HLE (A)	α/β	I: 23-193, 290-358 II: 194-289	I: 23-203, 290-358 II: 204-289	I: 27-170, 276-341 II: 1-26, 171-275

We now discuss our analysis of implementing the two-phase decomposition and amalgamation processes on chain A of protein 1CDG belonging to class β . Table – 4 summarizes the modularity value for each successive subdivisions of the protein contact network of 1CDG in the first phase of the approach. It may be noted that the fourth split with five clusters has the highest modularity value which then form input to the amalgamation phase. The result of aggregation is given in Table – 5. The first merging step with four domains has the highest modularity value and hence predicted as individual domains in protein 1CDG (A): D1 { C1 (503-518, 539-560), C5 (519-538, 561-584) = **503 – 584** }, D2 { C2 (1-142, 155-165, 213-234, 250-395) }, D3 { C3 (**396-502**) }, D4 { C4 (143-154, 166-212, 235-249, 585-688) }. The groups D2 and D4 seem to be interlinked from 1-395 and may hence be merged into a single domain, say, D2 = **1-395**, while the cluster of residues from 585-688 may form a domain, D4 = **585-688**, suggesting human judgment is necessary when clear boundaries are not obtained. The predicted result is now in good agreement with the CATH annotation (which is a manually curated database). No results were obtained from DomainParser for 1CDG (see Table – 6).

In Table – 7 predictions of domains using the modularity-based spectral graph approach discussed above for a representative set of proteins is compared with the domain annotation in the CATH database and the output of DomainParser. It may be noted that there is a reasonably good agreement not only in the number of domains predicted but also in the prediction of domain boundaries as compared to the annotation in CATH database.

Table 4. Subdivision of protein contact network of 1CDG (A)

Split No.	Modularity	No. of Clusters	Sequence Positions
1	0.433549	2	C: 143-154, 166-212, 235-249, 292-309, 396-688 C2: 1-142, 155-165, 213-234, 250-291, 310-395
2	0.575651	3	C1: 143-154, 166-212, 235-249, 503-688 C2: 1-142, 155-165, 213-234, 250-291, 310-395 C3: 292-309, 396-502
3	0.629698	4	C1: 503-584 C2: 1-142, 155-165, 213-234, 250-291, 310-395 C3: 292-309, 396-502 C4: 143-154, 166-212, 235-249, 585-688
4	0.638351	5	C1: 503-518, 539-560 C2: 1-142, 155-165, 213-234, 250-395 C3: 396-502 C4: 143-154, 166-212, 235-249, 585-688 C5: 519-538, 561-584
5	0.625443	6	C1: 503-505, 514-518, 539-551 C2: 1-142, 155-165, 213-234, 250-395 C3: 396-502 C4: 143-154, 166-212, 235-249, 584-688 C5: 519-538, 561-579 C6: 506-513, 552-560, 580-583

Table 5. Amalgamation of protein cluster graph of protein1CDG (A)

Merge No.	Modularity	No. of Clusters	Agglomeration of Clusters
1	0.227604	4	D1: C1, C5 D2: C2 D3: C3 D4: C4
2	0.0221965	3	D1: C1, C5 D2: C2, C3 D3: C4
3	-0.131987	2	D1: C1, C2, C3, C5 D2: C4

In Table – 7 predictions of domains using the modularity-based spectral graph approach discussed above for a representative set of 11 more proteins is compared with the domain annotation in the CATH database and the output of DomainParser. It may

Table 6. Comparison of CATH annotation, DomainParser results with our spectral method

PDB ID	Class	CATH Results	DomainParser Results	Spectral Graph Method
1CDG (A)	Mainly β	I: 1-400 II: 401-495 III: 496-582 IV: 582-685	Error Message: Cannot Parse 1cdg A	I: 1-395 II: 396-502 III: 503-584 IV: 585-688

Table 7. Domain prediction by spectral graph approach, CATH and DomainParser

PDB ID	Class	CATH Results	DomainParser Results	Graph Spectral Method
1PPR (M)	α	I: 1-155 II: 156-312	I: 1-156, 198-216, 262-277 II: 157-178, 217-261, 299-312	I: 1-25, 75-155 II: 26-74, 156-312
1UGO (A)	α	I: 1-99	I: 13-99	I: 20-99 II: 1-19(<30, ignore)
1PGR (B)	β	I: 1-108 II: 109-213	I: 1-108 II: 109-213	I: 1-108 II: 109-209
1CDG (A)	Mainly β	I: 1-400 II: 401-495 III: 496-582 III: 582-685	Error Message: Cannot Parse 1cdg A	I: 1-395 II: 396-502 III: 503-584 III: 585-688
2BW4 (A)	B	I: 7-158 II: 159-328	I: 2-310	I: 47-166 II: 167-375
1KF6 (B)	$\alpha\beta$	I: 2-105 II: 106-243	I: 1-107 II: 108-243	I: 1-89 II: 90-243
16PK (A)	$\alpha\beta$	I: 5-192 II: 199-406	Error Message: Cannot parse 16pk A	I: 1-197 II: 198-404
1MSH (A)	$\alpha+\beta$	I: 1-72	I: 1-72	I: 1-72
1D5M (B)	$\alpha+\beta$	I: 4-81 II: 82-181	I: 2-92 II: 93-190	I: 1-86 II: 87-181
1HLE (A)	$\alpha\beta$	I: 23-193, 290-358 II: 194-289	I: 23-203, 290-358 II: 204-289	I: 26-170, 276-341 II: 1-26, 171-275
1LAM (A)	$\alpha\beta$	I: 1-161 II: 162-484	I: 1-162 II: 163-484	I: 1-161 II: 162-484
1SMP (A)	β	I: 1-17, 250-471 II: 18-249	I: 4-15, 250-374 II: 16-56, 183-249 III: 375-471 III: 57-182	I: 1-12, 250-468 II: 13-249
5PEP (A)	β	I: 1-170 II: 171-327	No. of Domains: 1 Cannot Parse 5pep A	I: 1-185 II: 186-306

be noted that there is a very good agreement not only in the number of domains predicted but also in the prediction of domain boundaries as compared to the annotation in CATH database.

4 Conclusion

Here we proposed a simple and elegant graph-based approach for domain identification which not only decomposes the protein into domains but also assesses the quality of the decomposition. The approach involves two steps. In the first step Newman's modularity approach for partitioning a graph into clusters is implemented and the leading eigenvectors of the modularity matrix analyzed. The split with highest value of a quality function, called modularity, which is defined as the difference in the number of edges falling within groups and the expected number in an equivalent network with edges placed at random, is used for terminating the division of the graph. These clusters are then used to construct a graph with each cluster as node and edges drawn between them based on the interactions between the elements of the cluster. The aggregation approach by Clauset *et al* implemented on this structure graph helps in identifying unique structural domains in proteins. The prediction is in good agreement with the annotation in the manually curated CATH database for most examples shown here. However, we do observe that for some proteins clear boundaries are predicted by the approach discussed here, suggesting the need for human judgment. DomainParser is also based on constructing a network model with each atom defined as a node and edges drawn if two atoms are within 4 Å distance. Compared to DomainParser, our protein network is constructed using each C_α atom as node thereby reducing the complexity of the network. Even with this simplification, the prediction by our approach is, in general, in better agreement with annotation in CATH compared to that of DomainParser. The DomainParser algorithm partitions a protein structure into domains accurately when the number of domains to be partitioned is known. However the performance drops when this number is unclear. No such prior information is required in this approach. The modularity based approach discussed here identifies not only two or more domains in a protein, but also identifies domains that are comprised of residues not contiguous along the polypeptide chain.

References

1. Janin, J., Wodak, S.J.: Structural domains in proteins and their role in the dynamics of protein function. *Prog. Biophys. Mol. Biol.* 42, 21–78 (1983)
2. 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 Research* 28, 235–242 (2000)
3. Swindells, M.B.: A procedure for the automatic determination of hydrophobic cores in protein structures. *Protein Sci.* 4, 93–102 (1995)
4. Holm, L., Sander, C.: The FSSP database of structurally aligned protein fold families. *Nucl. Acid Res.* 22, 3600–3609 (1994)
5. Siddiqui, A.S., Barton, G.J.: Continuous and discontinuous domains, an algorithm for the automatic generation of reliable protein domain definitions. *Protein Sci.* 4, 872–884 (1995)

6. Xu, Y., Xu, D., Gabow, H.: Protein domain decomposition using a graph-theoretic approach. *Bioinformatics* 16, 1091–1104 (2000)
7. Ramesh, K., Sistla, B.K.V., Vishveshwara, S.: Identification of Domains and Domain Interface Residues in Multidomain Proteins From Graph Spectral Method. *Structure, Function, and Bioinformatics* 59, 616–626 (2005)
8. Jones, S., Stewart, M., Michie, A., Swindells, M.B., Orengo, C., Thornton, J.M.: Domain assignment for protein structures using a consensus approach, characterization and analysis. *Protein Science* 7, 233–242 (1998)
9. Conte, L.L., Ailey, B., Hubbard, T.J.P., Brenner, S.E., Murzin, A.G., Chothia, C.: SCOP: a structural classification of protein database. *Nucleic Acid Res.* 28, 257–259 (2000)
10. Veretnik, S., Bourne, P.E., Alexandrov, N.N., Shindyalov, I.N.: Toward consistent assignment of structural domains in proteins. *J. Mol. Biol.* 339, 647–678 (2004)
11. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev.* 74 (2006), id. 036104
12. Newman, M.E.J.: Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA.* 103, 8577–8582 (2006)
13. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. *Phys. Rev.* 70 (2004), id. 066111
14. Holm, L., Sander, C.: Parser for protein folding units. *Proteins* 19, 256–268 (1994)
15. Ford, L.R., Fulkerson, D.R.: *Flows in Networks*. Princeton University Press, Princeton (1962)
16. Csardi, G., Nepusz, T.: The igraph software package for complex network research. *Inter-Journal Complex Systems.* 1695 (2006)

Author Index

- Agrawal, Ganesh Kumar 18
Aiguier, Marc 270
Al Seesi, Sahar 90
Altiparmak, Fatih 424
Aluru, Srinivas 30
Ammar, Reda 90
Angibaud, Sébastien 102
Atluri, Gowtham 1

Banks, Eric 14
Bansal, Mukul S. 114
Beerenwinkel, Niko 362
Bollenbeck, Felix 126
Boucher, Christina 139
Bozdağ, Doruk 151, 320
Brown, Daniel G. 139

Camerlengo, Terry 320
Cao, Wei 164
Carmichael, Ted 187
Carlborg, Örjan 307
Catalyurek, Umit V. 151, 320
Chang, Wen-Chieh 114
Chazelle, Bernard 14
Cooper, Nigel G.F. 353
Crooks, Lucy 307
Cui, Chun-Ying 199

Dai, Yang 211
Davis, Joseph E. 176
Dong, Difeng 199
Dréau, Didier 187
Dunker, A. Keith 18
Dutt, Shantanu 211

Emrich, Scott J. 343
Erten, Cesim 224
Eulenstein, Oliver 114
Eveillard, Damien 102

Fang, Gang 1
Ferhatosmanoglu, Hakan 424
Fernández-Baca, David 114
Fertin, Guillaume 102
Filkov, Vladimir 16
Fontanarosa, Joel 211
Fujan, Scott 409

Gao, Jianjiong 18
González, Alvaro J. 236
Graham, James H. 353
Gupta, Rohit 1

Hadzikadic, Mirsad 187
Hale, John 409
Hartley, Tim 320
Hérisson, Joan 270
Holmgren, Sverker 307
Hsu, Sam 259
Hu, Jianjun 248
Huang, Kun 320
Huang, Tim 320
Huang, Yi-Wen 320

Imoto, Seiya 54

Jackson, Benjamin G. 30
Junier, Ivan 270

Kamal, Abu H.M. 259
Kaspar, Stephanie 126
Képès, François 270
Khokhar, Ashfaq 362
Kitamoto, Katsuhiko 164
Kumar, Vipin 1
Kuplicki, Rayus 409

Lam, T.W. 79
Le Gall, Pascale 270
Li, Weiming 44
Liao, Li 236

Ma, Bin 44
Ma, Xiaoping 295
Magnin, Morgan 282
Mahgoub, Imad 388
Manceney, Matthieu 270
Măndoiu, Ion 52
Maurin, Mylène 282
Miyano, Satoru 54
Mock, Hans-Peter 126
Mow, Benjamin 199
Murali, T.M. 67

- Nabieva, Elena 14
 Nagasaki, Masao 54
 Nakamura, Shugo 164
 Nan, Fangyuan 295
 Nettelblad, Carl 307

 Obradovic, Zoran 18
 Olson, Michael 343
 Ozer, Hatice Gulcin 320
 Ozsoy, Adnan 176

 Pandey, Gaurav 1
 Pandya, Abhijit S. 259
 Parekh, Nita 437
 Park, Hyun Jung 331
 Parthasarathy, Srinivasan 424
 Parvin, Jeffrey D. 151, 320
 Patel, Sandeep 176
 Peterson, Ryan 14

 Rajasekaran, Sanguthevar 90
 Regier, Allison 343
 Ren, Huan 211
 Rivera, Corban G. 67
 Rouchka, Eric C. 353
 Roux, Olivier 282
 Rusu, Irena 102

 Saeed, Fahad 362
 Sarratt, Trevor 409
 Schnable, Patrick S. 30
 Seiffert, Udo 126
 Shi, Xinghua 376
 Shimizu, Kentaro 164
 Shoaib, Muhammad 259, 388
 Singh, Mona 14

 Slavik, Michael 388
 Smieja, Jaroslaw 400
 Sözdinler, Melih 224
 Stanimirov, Dimitre 187
 Steinbach, Michael 1
 Stevens, Rick 376
 Sumikoshi, Kazuya 164
 Sung, Wing-Kin 79

 Tamada, Yoshinori 54
 Taufer, Michela 176
 Terada, Tohru 164
 Thelen, Jay J. 18
 Tyree, Stephen 409

 Ucar, Duygu 424

 Wang, Xiangping 353
 Wang, Yaonan 295
 Weier, Diana 126
 Williams, Tiffani L. 331
 Wong, Limsoon 199
 Wong, Thomas K.F. 79
 Wu, Jiejun 320

 Xu, Dong 18

 Yalamanchili, Hari Krishna 437
 Yamaguchi, Rui 54
 Yiu, S.M. 79

 Zagordi, Osvaldo 362
 Zhang, Fan 248
 Zhang, Kaizhong 44
 Zhu, Xingquan 259, 388