Mark Borodovsky
Johann Peter Gogarten
Teresa M. Przytycka
Sanguthevar Rajasekaran (Eds.)

Bioinformatics Research and Applications

LNBI 6053

6th International Symposium, ISBRA 2010
Storrs, CT, USA, May 2010
Proceedings

Springer

Lecture Notes in Bioinformatics 6053

Subseries of Lecture Notes in Computer Science

Mark Borodovsky Johann Peter Gogarten
Teresa M. Przytycka Sanguthevar Rajasekaran
(Eds.)

Bioinformatics Research and Applications

6th International Symposium, ISBRA 2010
Storrs, CT, USA, May 23-26, 2010
Proceedings

Springer

Volume Editors

Mark Borodovsky
Georgia Institute of Technology
School of Biology
Atlanta, Georgia, USA
E-mail: mark.borodovsky@biology.gatech.edu

Johann Peter Gogarten
University of Connecticut
Molecular and Cell Biology Department
Storrs, CT, USA
E-mail: gogarten@uconn.edu

Teresa M. Przytycka
National Center for Biotechnology Information
National Library of Medicine
National Institutes of Health
Bethesda, MD, USA
E-mail: przytyck@ncbi.nlm.nih.gov

Sanguthevar Rajasekaran
University of Connecticut
Department of Computer Science and Engineering
Storrs, CT, USA
E-mail: rajasek@engr.uconn.edu

Library of Congress Control Number: Applied for

CR Subject Classification (1998): H.3, J.3, H.2.8, H.4, F.1, I.5

LNCS Sublibrary: SL 8 – Bioinformatics

ISSN 0302-9743
ISBN-10 3-642-13077-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13077-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

The 6^{th} International Symposium on Bioinformatics Research and Applications (ISBRA 2010) was held during May 23–26, 2010 at the University of Connecticut, Storrs, Connecticut. The symposium provided a forum for the exchange of new results and ideas among researchers, developers, and practitioners working on all aspects of bioinformatics, computational biology, and their applications.

The program of the symposium included 20 contributed papers selected by the Program Committee from 57 submissions received in response to the call for papers. The symposium also included poster presentations and featured invited keynote talks by six distinguished speakers: Catalin Barbacioru from Life Technologies spoke on tracing the early cell divisions of mouse embryos by single cell RNA-seq, Piotr Berman from Pennsylvania State University spoke on successes and failures of elegant algorithms in computational biology, Mark Gerstein from Yale University spoke on human genome annotation, Ivan Ovcharenko from the National Center for Biotechnology Information spoke on the structure of proximal and distant regulatory elements in the human genome, Laxmi Parida from the IBM T.J. Watson Research Center spoke on combinatorics in recombinational population genomics, and Mona Singh from Princeton University spoke on predicting and analyzing cellular networks.

We would like to thank the Program Committee members and external reviewers for volunteering their time to review and discuss symposium papers. We would like to extend special thanks to the Steering and General Chairs of the symposium for their leadership, and to the Finance, Publicity, Local Organization, and Posters Chairs for the hard work in making ISBRA 2010 a successful event. Last but not least we would like to thank all authors for presenting their research work at the symposium.

May 2010

Mark Borodovsky
J. Peter Gogarten
Teresa Przytycka
Sanguthevar Rajasekaran

Symposium Organization

Steering Chairs

Dan Gusfield	University of California, Davis, USA
Yi Pan	Georgia State University, USA
Marie-France Sagot	INRIA, France

General Chairs

Ion Măndoiu	University of Connecticut, USA
Alexander Zelikovsky	Georgia State University, USA

Program Chairs

Mark Borodovsky	Georgia Institute of Technology, USA
Peter Gogarten	University of Connecticut, USA
Teresa Przytycka	National Institutes of Health, USA
Sanguthevar Rajasekaran	University of Connecticut, USA

Publicity Chair

Dumitru Brinza	Life Technologies, USA

Finance Chairs

Anu Bourgeois	Georgia State University, USA
Raj Sunderraman	Georgia State University, USA

Workshop and Poster Chairs

Yufeng Wu	University of Connecticut, USA
Craig E. Nelson	University of Connecticut, USA

Local Organization Chairs

Reda A. Ammar	University of Connecticut, USA
Les Loew	University of Connecticut Health Center, USA
Linda Strausbaugh	University of Connecticut, USA

Program Committee

Richa Agarwala	NCBI
Srinivas Aluru	Iowa State University
Danny Barash	Ben-Gurion University
Robert Beiko	Dalhousie University
Anne Bergeron	Universite du Quebec a Montreal
Daniel Berrar	University of Ulster
Olivier Bodenreider	NLM
Paola Bonizzoni	Universitá Degli Studi di Milano-Bicocca
Mark Borodovsky	Georgia Tech (Co-chair)
Daniel Brown	University of Waterloo
Tien-Hao Chang	National Cheng Kung University
Chien-Yu Chen	National Taiwan University
Luonan Chen	Osaka Sangyo University
Bhaskar DasGupta	University of Illinois at Chicago
Amitava Datta	University of Western Australia
Guillaume Fertin	University of Nantes
Vladimir Filkov	University of California Davis
Jean Gao	University of Texas at Arlington
J. Peter Gogarten	University of Connecticut (Co-chair)
Katia Guimaraes	Federal University of Pernambuco
Jieyue He	Southeast University
Manuela Helmer-Citterich	Univ. “Tor Vergata” Roma
Vasant Honavar	Iowa State University
Jinling Huang	Eastern Carolina University
S. Iyengar	Louisiana State University
Lars Kaderali	University of Heidelberg
Ming-Yang Kao	Northwestern University
Yury Khudyakov	CDC
Danny Krizanc	Wesleyan University
Jing Li	Case Western Reserve University
Guohui Lin	University of Alberta
Fenglou Mao	University of Georgia
Osamu Maruyama	Kyushu University
Satoru Miyano	University of Tokyo
Ion Moraru	University of Connecticut Health Center
Bernard Moret	EPFL
Giri Narasimhan	Florida International University
Andrei Paun	Louisiana Tech University
Itsik Pe’er	Columbia University
Mihai Pop	University of Maryland
Maria Poptsova	University of Connecticut
Teresa Przytycka	NCBI (Co-chair)
Sven Rahmann	Technical University Dortmund
Sanguthevar Rajasekaran	University of Connecticut (Co-chair)
Shoba Ranganathan	Macquarie University

S. Cenk Sahinalp	Simon Fraser University
Sartaj Sahni	University of Florida
David Sankoff	University of Ottawa
Russell Schwartz	Carnegie Mellon University
Joao Setubal	Virginia Bioinformatics Institute
Mona Singh	Princeton University
Steven Skiena	Stony Brook University
Jens Stoye	Bielefeld University
Raj Sunderraman	Georgia State University
Wing-Kin Sung	Nuational University of Singapore
Sing-Hoi Sze	Texas A&M University
Haixu Tang	Indiana University
Gabriel Valiente	Technical University of Catalonia
Jean-Philippe Vert	Ecole des Mines de Paris
Stéphane Vialette	Université Paris-Est Marne-la-Vallée
Li-San Wang	University of Pennsylvania
Lusheng Wang	City University of Hong Kong
Carsten Wiuf	Aarhus University
Richard Wong	Kanazawa University
Yufeng Wu	University of Connecticut
Yanqing Zhang	Georgia State University
Leming Zhou	University of Pittsburgh

External Reviewers

Ardalan, Adel
Bernauer, Julie
Bianchi, Valerio
Bourdon, Jeremie
Chacko, Elsa
Dao, Phuong
Ding, Zejin
Dondi, Riccardo
Fang, Ming
Gabdank, Idan
Gerlach, Wolfgang
Georgi, Benjamin
Hoffmann, Nils
Hormozdiari, Farhad
Kiani, Narsis Aftab Kim, Dongchul
Kumar, Gaurav
Lee, Byoungkoo
Li, Fan
Li, Xin
Lin, Yu
Liu, Kevin
Ma, Qin
Mao, Xizeng
Mayampurath, Anoop
Parca, Luca
Pyon, YoonSoo
Rendon, David Stiven Campo
Reyaz-Ahmed, Anjum
Rizzi, Raffaella
Rusu, Irena
Sikora, Florian
Sperschneider, Jana
Subramanian, Ayshwarya
Tsai, Ming-Chi
Wittler, Roland
Wolf, Thomas
Yang, Zefeng
Yiu, S.M.
Yorukoglu, Deniz
Zhang, Xiuwei
Zhou, Fengfeng

Table of Contents

Tracing the Early Cell Divisions of Mouse Embryos by Single Cell RNA-Seq
(Invited Keynote Talk)

Catalin Barbacioru

Genetic Systems, Life Technologies
850 Lincoln Centre Dr, Foster City, CA 94404, USA
catalin.barbacioru@lifetech.com

The identity and function of a cell is determined by its entire RNA component, which is called the transcriptome of a cell. The transcriptome is the functional readout of the genome and epigenome. In an organism, essentially every cell has the same genome, while every cell type and potentially each individual cell has a unique transcriptome. Ideally, the transcriptome analysis should capture the exact quantity of all full length RNAs of all classes at single-base resolution in the smallest functional unit of an organism, an individual cell.

RNA-Seq has been recently employed to characterize the transcriptome of several human, mouse and fly tissues and cells. These studies indicate that RNA-Seq is a highly specific and sensitive technique that can discover low expressed genes, down to a single RNA copy per cell. With the development of single-cell application we were able to sequence 52 single mouse blastomeres at different development stages, representing oocyte cells, 2-cell, 4-cell, 8-cell stages, inner cell mass, trophectoderm and embryonic stem cells.

While the expression profiles between blastomeres within the same 2-cell embryo are similar to each other but not identical, there are more than one thousand genes up/down regulated between blastomeres at 4-cell and 8-cell embryo stages, which correspond to the second wave of zygotic genome activation. These changes are rarely reproducible between embryos, suggesting that cells at these early developmental stages, in their search for final lineage diversification, can be randomly initiated from the newly expressed genes in each cell stage. Changes in gene expression are complemented with structural changes. Significant changes in 3UTR coverage are observed between different cell embryo stages, a shortening of the 3UTR being noticed in oocyte and in latter developmental stages. Additionally, we observed significant differences in abundance of repetitive elements between different cell types, including some associations with the shortening of the 3'UTR.

We created an automated alignment pipeline that produces expression profiling of annotated features, alternative splicing detection and polyadenylation sites identification. Alignment tools were combined with statistical approaches taking into account sequence dependent variations and allowed us to perform various comparisons within and between cell types.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, p. 1, 2010.

Successes and Failures of Elegant Algorithms in Computational Biology
(Invited Keynote Talk)

Piotr Berman

Department of Computer Science and Engineering
Penn State
University Park, PA 16802
berman@cse.psu.edu

Problems that originated in biology gave rise to many very nice computational problems, and they motivated a large body of research, and many very elegant results. But are these results useful in biology?

The record is mixed, and we will review both successes and failures.

Our examples will include applications of set cover and tiling problems and problems related to biological networks. In some cases, new algorithms provided biologists with efficient solutions to their problems, in other, not so much, as the complex nature of the motivating problems was lost in the translation into the language of algorithmic problems.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, p. 2, 2010.

Modeling without Borders: Creating and Annotating VCell Models Using the Web

Michael L. Blinov, Oliver Ruebenacker, James C. Schaff, and Ion I. Moraru

Center for Cell Analysis and Modeling
University of Connecticut Health Center
Farmington, CT 06030, USA
{blinov,oruebenacker,schaff,moraru}@exchange.uchc.edu
http:vcell.org/sybil

Abstract. Biological research is becoming increasingly complex and data-rich, with multiple public databases providing a variety of resources: hundreds of thousands of substances and interactions, hundreds of ready to use models, controlled terms for locations and reaction types, links to reference materials (data and/or publications), etc. Mathematical modeling can be used to integrate this complex data and create quantitative, testable predictions based on the current state of knowledge of a biological process. Data retrieval, visualization, flexible querying, and model annotation for future reuse, are some of the important requirements for modeling-based research in the modern age. Here we describe an approach that we implement within the popular Virtual Cell (VCell) modeling and simulation framework in order to help connect the modeling community with the web of machine-processable systems biology knowledge. A new software application, called SyBiL (Systems Biology Linker), has been designed and developed for simultaneous querying of multiple systems biology knowledge bases and data sources, such as web repositories, databases, and user files, and converting the extracted and refined data into model elements. Integration of SyBiL as a component of VCell makes these capabilities easily available to a wide modeling community.

Keywords: Biological databases, mathematical modeling, VCell, data conversion.

1 Introduction

Mathematical modeling is increasingly necessarry to investigate the function of molecular pathways and networks, and a growing number of resources that facilitate computational approaches are becoming available. On the one hand, a large collection of public databases can help researchers gather existing knowledge about molecular interactions: databases such as Reactome [1], the BioCyc collection of Pathway/Genome databases [2], Pathway Interaction Database (PID) [3], BioModels repository of computable models [4], Integrating Network Objects with Hierarchies (INOH) database [5], Kyoto Encyclopedia of Genes and

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 3–17, 2010.

Genomes [6] store information about thousands of molecular species and their interactions. On the other hand, a lot of software tools can help researchers to create and simulate mathematical models of such interactions: a few examples of the more widely used platforms are VCell [7], [8], Copasi [9], CellDesigner [10]. However, formats that are typically used to store molecular pathway information (e.g. Biological Pathway Exchange standard, BioPAX, [11]) are database-centric and formats used by modeling software (e.g. Systems Biology Markup Language, SBML, [12]) are simulation-centric, with significant semantic and structural differences. This makes exchange of information between databases and modeling tools problematic. There are several tools that provide conversion from BioPAX to SBML. Each of them has distinct advantages for particular uses. BiNoM [13] is a plugin for Cytoscape [14], a powerful tool for analysis of data represented as graphs with rich data representation and visualization (primary graph layouts and scaling) capabilities. Paxtools library [15] is the primary tool for working with BioPAX format and provides excellent capabilities for analysis of BioPAX data. Some pathway databases (e.g. Reactome, INOH) now provide output in the form of SBML file. However, none of these tools is truly modeling-oriented.

In this manuscript we describe a framework, called **S**ystems **Bi**ology **L**inker, **SyBiL** [16], [17], that is designed to assist a modeler in using data from multiple online sources. The unique strength of the proposed framework is a model-oriented approach, which can handle issues that can not be tackled either by current tools such as Cytoscape and PaxTools, or by straightforward export/format conversion from databases. A few examples are the capabilities to query a variety of resources from a modeling point of view, such as:

- What database entities can be used as species?
- Should a subset or superset of database entities be used as species in the model?
- Should variants of a given protein (e.g. different phosphoforms, different conformational states) be made into distinct species?
- Should multiple ligands be converted into a single or multiple species?
- Can we keep track of all such assumptions, so that when future changes are made (like dividing a single species corresponding to all ligands into several species), this can be easily done?

In Section 2 below we analyze the multiple problems faced by researchers when attempting to create mathematical models based on the knowledge from pathway databases. In Section 3 we describe the approach to integrate data sources into a modeling process. In Section 4 we describe a framework for querying multiple resources to use in model building. In Section 5 we discuss how this framework will be incorporated into the Virtual Cell (VCell) modeling and simulation platform in order to provide an expert system guiding users in building and annotating computational models. Finally, in section 6 we talk about future plans and problems that we are facing implementing our approach.

2 Challenges for Building Quantitative Models of Molecular Interaction Networks

2.1 Accessing Multiple Systems Biology Resources

Progress in systems biology crucially depends on the capacity to share machine-processable knowledge (knowledge data), which is the product of costly curation. The benefits of knowledge data manifest in many ways, from more powerful querying to new approaches in data analysis to building and annotating models. This has spawned a diverse multitude of knowledge bases on the web offering free knowledge data about substances, interactions, pathways, models, anatomical features and literature references.

Sharing knowledge data depends on standardized formats which typically result from sustained community efforts around specific interests such as BioPAX for pathways and SBML for models. While the domains of these still evolving standards overlap (e.g. both BioPAX and SBML represent substances and their interactions), conversion usually results in loss of information.

Access protocols differ naturally between knowledge bases. While downloading a data set in a popular format may be a similar experience across knowledge bases, the interface of the query and selection capabilities differs widely. Knowledge data from various sources has been aggregated by Pathway Commons [18] and Bio2RDF [19], but such aggregation necessarily lags behind the original. Also, both projects change the original modifiers and require considerable technical skills to be used efficiently, which prevent most users from using them. Moreover, neither makes use of SBML models.

More problematically, when delivery and end use of data are divorced, the mode of retrieval may be poorly adjusted to the requirements of the use. In the worst case, a user will need three different tools for retrieval, evaluation and final use of the data. If however, a single tool is responsible for retrieval and final use, it can help the user decide what data is needed and where it can be obtained, and it can filter retrieved data.

If the entire online knowledge data was stored in a single base, in a single format and with a single access protocol, a simple query could yield all available data. In reality, we have different knowledge bases, formats and protocols to serve different priorities, and the best combination of base, format and protocol depends on the given interest. Some interests are best served by a specific combination of source, format and protocol: for example, to find a model, we can query BioModels database and obtain it in SBML, or to find a pathway, we can query Pathway Commons and obtain it in BioPAX, or to find a protein, we can enter its name in UniProt database [20] and get a listing.

Other interests are best served by querying multiple sources, which requires integrating multiple formats and protocols. For example, if the interest is to find substances with which EGFR interacts, candidates are found in BioModels, Pathway Commons and UniProt. Interfaces differ and results are available in SBML, BioPAX and site-specific format respectively. Moreover, the interest does not require entire models or pathways, but only fractions thereof. Checking out

multiple sources separately and submitting a query to each, using the necessary formats and protocols, is an enormous task, especially since sources, formats and protocols evolve over time.

At the same time, efficient queries exploit the fact that resources are extensively interlinked. For example, to investigate EGFR, a user typically would start with the name EGFR, locate a UniProt identifier (for the desired organism) and then use the identifier to find models and pathways. This can be easily automated. What is needed therefore is a tool that accepts, for example, a protein name and an organism, and generates a list of possible reaction partners. This involves looking up identifiers, submitting queries simultaneously to multiple sources, supporting multiple formats and multiple protocols, and gathering the results into a single list. It requires an integration of multiple formats and multiple protocols.

2.2 Model-Driven Use of Pathway Data Requires Human Decision Making

In a mathematical model, whether two compounds constitute the same or different substances usually depends on whether they behave differently or not in that particular model scenario. Knowledge present in databases, however, aims to make statements about entities independent of any particular scenario. Due to these differences, many elements do not map one-to-one between models and pathway data.

- A species in a model can be a subset of a database entity. For example, Epidermal Growth Factor receptor protein is typically a single entity in a database, but some mathematical models have species that correspond to multiple phosphoforms of it [21].
- A species in a model can represent two or more database entities. For example, multiple isoforms of a protein represented by distinct database entities can be lumped into a single species [21].

As we see, sometimes the question of whether a species will be converted to one or more physical entities, and vice versa, is not trivial. In [16] we discuss how this question can be answered automatically by extensive analysis of extensions and annotations in the source file or import from other sources. While most common cases can be automated, we pointed out that a few cases remain where user intervention might be necessary [22], such as determining the topology of locations, substituting a substance with another one that is equivalent in a particular model, or introducing certain assumptions into a model.

2.3 Recording Modeling Assumptions

As we have seen above, a conversion between database information and models requires certain human decisions. These decisions have to be stored in some format, so it can be reused in future data manipulations (reverse conversions,

merging with other data, etc.). We need a systematic approach for storing and reusing these relationships and thus making conversions between knowledge and model reproducible and reversible. We introduced a bridging ontology, SBPAX (Systems Biology Pathway Exchange), described in [22] and summarized in section 3. The SBPAX bridging ontology allows the conversion between multiple formats (modeling and database) to be performed as two consecutive one-to-one mappings, with an intermediate refinement step that is performed on, and recorded in, the SBPAX data.

2.4 Merging Models

Even the well-annotated models stored in BioModels repository that are compliant with MIRIAM (Minimum information requested in the annotation of biochemical models) standard present challenges [23], [24]. For example, currently there is no standard way in SBML to distinguish between different states of the same molecules. Thus, all phosphoforms of the same receptor will be annotated with the same reference identifier (GO term from Gene Ontology [25], UniProt key, etc.). This means that there still will be often the case that it is impossible to automatically tell whether species X of model A and species Y of model B, which have the same reference identifier, are indeed identical and should be mapped into the same species in a merged model. Linking each species to the (multiple) sources of knowledge and recording assumptions on how these sources were used can greatly facilitate automatic merging of models.

2.5 Using Models to Appreciate the Complexity of Underlying Biology

Each model is usually based on a selection of a small part of a larger network of interactions. However, this point is often missing when a model is presented [26]. Linking each model element to biological knowledge will provide an opportunity to create model-based knowledge bases. In such a case, when logging in to a public model, for example, clicking on a component of interest brings up a battery of potential modifications, interactions, and activities, and the likelihoods and potential consequences of each under a variety of typical sets of conditions, or specific conditions set by the user.

3 Infrastructure for Data-Driven Modeling

3.1 SBPAX Bridging Ontology

To support the integration of datasets related to molecular networks and pathways from different sources in different formats, we need to accomplish three tasks: converting data from one format to another, gluing corresponding data sets in different formats, and merging multiple datasets into one. The core of the underlying technology we developed to accomplish these tasks is an RDF/OWL-based ontology called SBPAX, short for Systems Biology Pathway Exchange [22].

It is designed to integrate multiple formats and based on a Web Ontology Language OWL [27]. Advantage of using OWL include: (1) other ontology-based formats can be seamlessly integrated by providing relationships between their classes and properties and those of SBPAX; (2) other structured formats not based on OWL can be mapped one-to-one to OWL and linked after the mapping. This includes XML, relational databases or even comma-separated values.

SBPAX is designed to be able to express any molecular reaction network that can be expressed in SBML, BioPAX and formats similar to these. To accomplish this, SBPAX covers substances, processes and interactions, locations and stoichiometric coefficients, and hierarchies existing among these. In the case of an RDF/OWL-based format like BioPAX, data can be directly linked to SBPAX data. With other formats like SBML, data can be mirrored one-to-one to RDF/OWL-data in SBPAX that can be linked with other SBPAX terms. SBPAX data created by import from multiple sources can be exported to SBML, and every basic SBML term (e.g. species, reaction, species type or compartment) will contain, as annotation, the URI of its equivalent SBPAX object, which will be linked by relationships to all related terms in the imported files.

To give a simple example, a substance in SBPAX is defined as any group of molecules or other compounds. SBPAX provides properties to define a substance as a superset or subset of another substance, or as the union or intersection of two other substances (useful for substances defined by constraints, e.g. on their phosphorylation state). This way, we can create a substance hierarchy and identify substances that can cover more or less than one database entity.

3.2 Prototype of a Data-Driven Modeling Interface: Systems Biology Linker (SyBiL)

We have prototyped Systems Biology Linker (SyBiL) modeling framework [16], [17]. The initial version is geared towards work with two common standards BioPAX and SBML. It is designed to obtain, store and merge data in BioPAX format, and to facilitate generating of kinetic models expressed in SBML (http://vcell.org/sybil). The tool is providing a modeling access to complete range of BioPAX data, which is not essential for simulations, but valuable for understanding and reusing the models (such as organisms, different names, linking species to a variety of databases, etc.). It takes an advantage of easy visualization of BioPAX, where different BioPAX object classes (proteins, small molecules, complexes etc.) are represented by nodes differing in shape or color, and each object is linked to biological information from public databases. The SyBiL converts BioPAX data into a computable kinetic model in SBML. Remarkably, the model is first generated in SBPAX format and stores all the modeling decisions regarding conversion of BioPAX data into SBML. After conversion to SBML BioPAX data is used to provide each SBML species and interaction with a unique identifier.

To facilitate the organization of the data and to make selections, SyBiL provides a graphical interface that can handle any OWL data. OWL (Web Ontology Language) is designed for use by applications that need to process the content

of information instead of just presenting information to humans. OWL provides a framework for controlled vocabulary along with a formal semantics. Specific support is provided for the BioPAX ontology. Ontologies provide a great deal of flexibility in data representation, analysis and visualization, by organizing data in a way that allow intelligent selections, for example allowing the user to select which resources are visible and which are hidden, similar to the Cytoscape visualization framework. Furthermore, arbitrary sets of objects can be visually collapsed and expanded. The user can decide which kinds of property relationships are represented by graph edges. As the framework is based on the existing VCell software, it displays a graph consisting of interactions among BioPAX physical entities in the VCell style as a bipartite graph in its fully flattened form (with nodes for both physical entities and interactions). For each class of physical entities and interactions the framework provides a separate symbol. Each physical interaction is connected by an edge with the physical entities participating in it. Components of complexes can be shown. All other objects are hidden, until they are properties of an object which becomes selected.

4 Specific Querying of Multiple Resources for Model Building

We are developing SyBiL into a framework that will allow the modeler to get simultaneous access to most systems biology knowledge bases from within a modeling-oriented interface. By obtaining references through simple text queries or more advanced queries, and then getting related data from references, the user will get information sufficient to create and annotate model elements in one step.

Previously, SyBiL was able to create models from locally stored BioPAX files [16], [17], [22]. Recently, we added a new capability for generating models from data retrieved from Pathway Commons and UniProt web resources.

The interface forwards a keyword query to Pathway Commons. After entering a search string (EGFR) and clicking the search button, the matches are displayed as a tree (Fig. 1). Each match contains brief information (name, synonyms, excerpts, and organism) as well as a list of cross references and a list of pathways that include the matched entity. Since Pathway Commons returns multiple UniProt cross references for human EGFR without distinction, SyBiL then sends these cross references to UniProt, which lists P00533 as the one in use and the other ids as obsolete.

After a pathway is selected, it can be retrieved in the BioPAX format from Pathway Common by clicking on the “Get pathway” button. The entities contained in the pathway are presented as a list (Fig. 2), grouped into processes and various kinds of substances (DNA, RNA, proteins, metabolites and complexes). For example, selecting the protein EGFR_HUMAN imports from this pathway human EGFR as well as the complexes which it is part of and the interactions in this pathway in which EGFR or any of its complexes participate. The reaction network can be converted into a VCell model and then further edited and simulated.

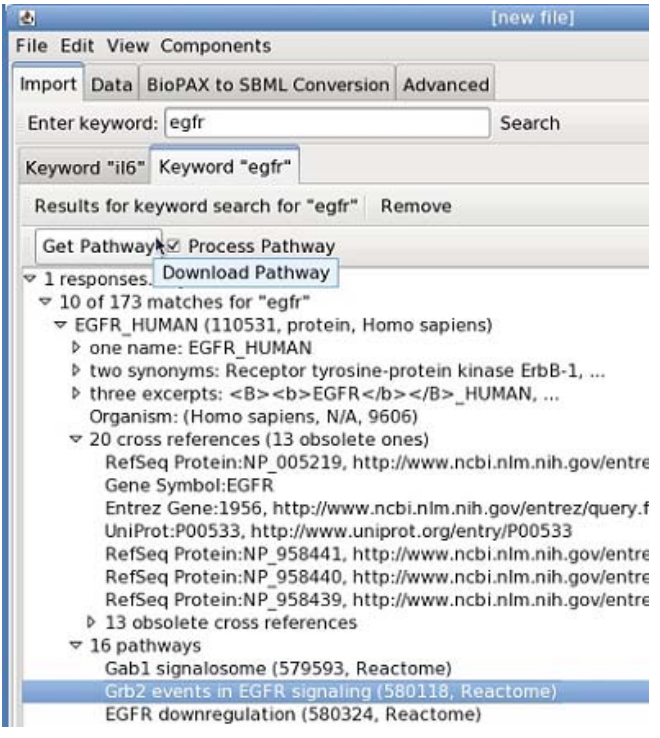


Fig. 1. Querying Pathway Commons for EGFR brings up all available information about entities that contain this term, presented as a tree. Under each protein entry there is a lot of extra oinformation, including a list of pathways that use this protein.

We plan to expand retrieval to data in other formats and from other resources, such as Bio2RDF, Uniprot Database, BioModels database, etc (Fig. 3) A user intending to use knowledge data to create, extend or annotate a mathematical model will be able to use the proposed interface as an entry point for selection of entities (e.g. substances and interactions) and their relationships to build his or her model.

The simplest starting point would be to enter a search term (such as a name of a protein) or a qualified identifier (e.g. UniProt P00533) to get knowledge data that can be used to create or annotate a species, as well as information about pathways, interactions and reaction partners for a name of interest. This information will be translated into an SBML model for use in any SBML-compliant software, or can be used by VCell software for modeling. A user will be able to send queries that rely on knowledge data, such as looking for components of a given complex, or participants in a given interaction, or find all entities at a given location, etc. A more advanced example of querying knowledge data is: given two substances, find the shortest chain (a list of substances such that each substance is a reaction partner for the substance next on the list) and connect them, under certain constraints. To eliminate trivial solutions, some constraint is needed. A

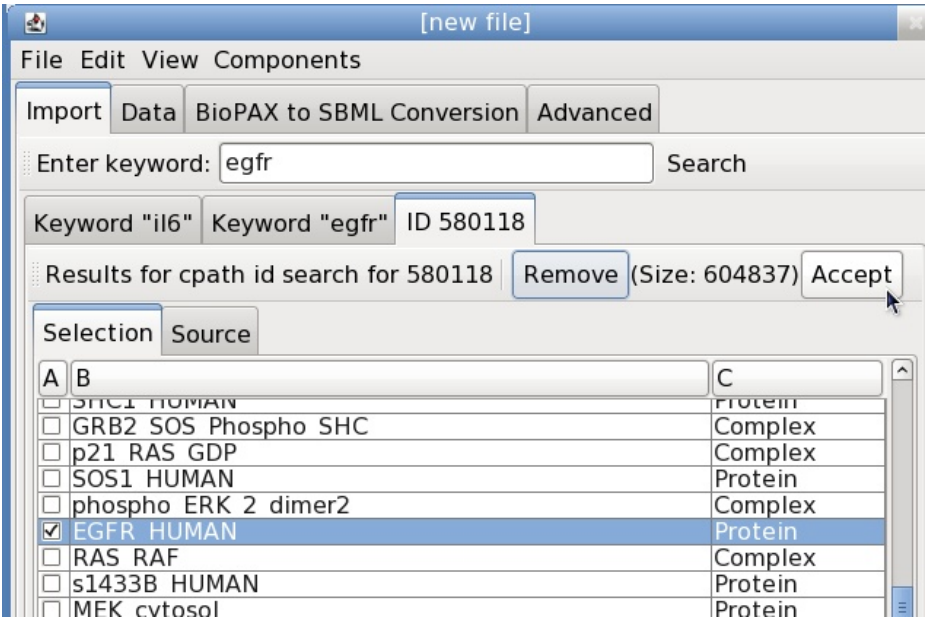


Fig. 2. Selecting a certain pathway brings up the number of entities that are used in this pathway. These entities itself can be selected to be imported into a modeling tool.

sensible constraint could be that every substance in a chain needs to be: (1) not ATP or (2) not a metabolite or (3) a protein. A reaction network will be built by aggregating related substances and interactions from multiple queries, where a substance or interaction obtained in one query can be the starting point for another query. The resulting network can be visualized with multiple levels of resolution. After the user selects substances and interactions and configures modeling assumptions, a mathematical model can be created and annotated automatically.

More specifically, the results of the generic query are presented as a raw SBPAX file (Fig. 4). It may include a variety of similar substances, such as EGFR, phosphorylated EGFR, EGFR phosphorylated at Y1174, EGFR unphosphorylated at Y1174, etc. For each substance, a modeler is provided with three basic pieces of information required to make a decision: location, chemical identity, and list of interactions. The user can set global options that determine which substances may be part of different subsets. This may depend on the system under consideration, but a typical setup would be that proteins, but not metabolites, can have different subsets. For example, this would mean that every interaction that involves ATP refers to the same substance, but different interactions may refer to different subsets of EGFR. After this global setup, the next step would be to go through the proteins and determine, for example, that EGF has no subsets, but EGFR has. BioPAX data usually contains information about the relevant post-translational modifications (as sequence features), which

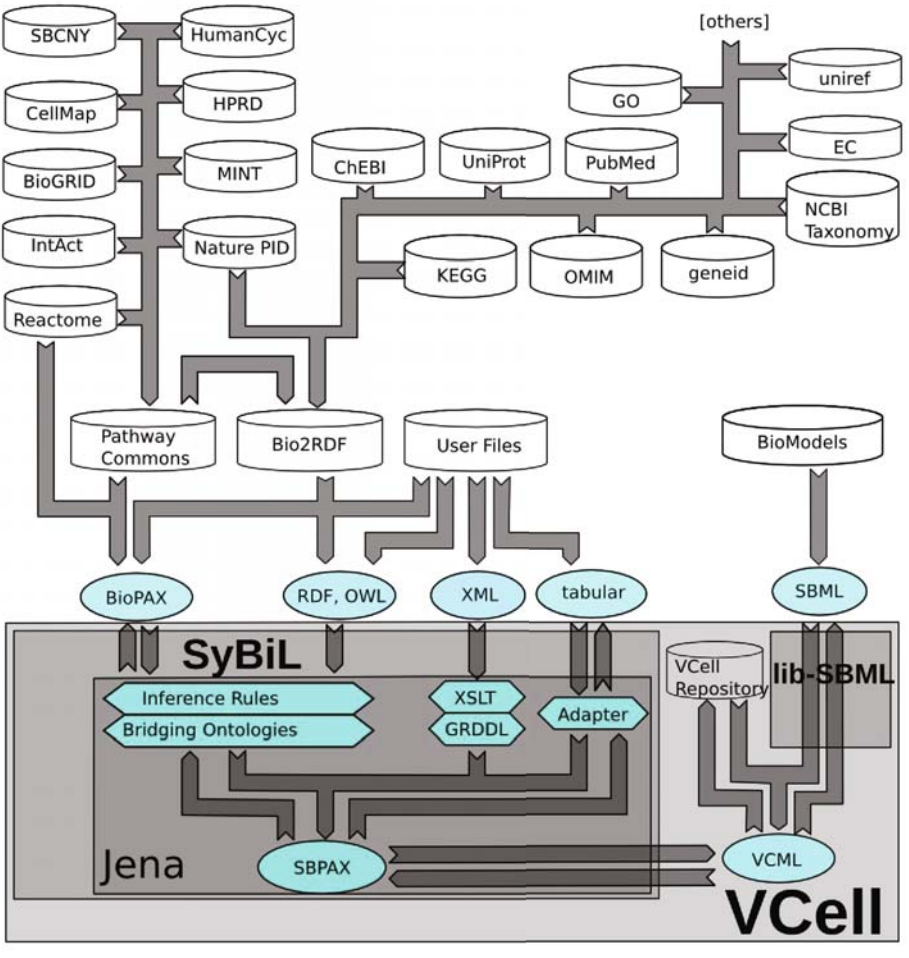


Fig. 3. Schematic description of SyBiL-VCell architecture. Arrows show the flow of knowledge from the knowledge bases (cylinders) to components of SyBiL and VCell and the formats used (ovals). Pathway Commons collects data from nine pathway bases available in BioPAX. Reactome's data in BioPAX is available from Pathway Commons, but can also be retrieved directly client. Bio2RDF collects knowledge data from a wide variety of sources, some shown here. The knowledge data collected by Bio2RDF is RDF which mirrors data that is either RDF or converted to RDF. Bio2RDF maps name spaces to internal name spaces which reveal the source of the data and can be trivially mapped back. SyBiL uses Jena to handle RDF and conversions from all above formats to SBPAX, by methods shown as hexagons. SyBiL in VCell allows conversions between SBPAX and VCML, and VCell handles conversions between VCML and SBML.

can be used to identify subsets of EGFR, which can be arranged into hierarchies in SBPAX. For example, EGFR phosphorylated at Y1174 is declared o be a subset of phosphorylated EGFR and subset of EGFR. After identifying these

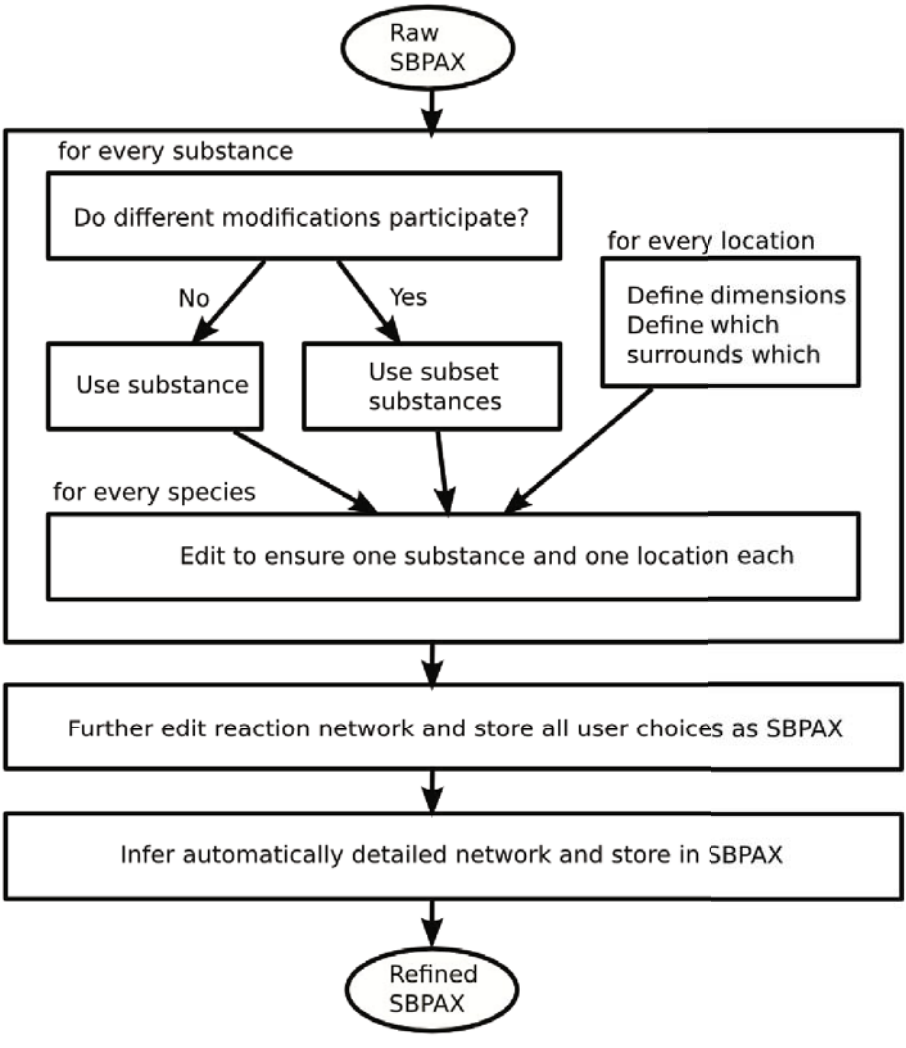


Fig. 4. Conversion of data into a model is performed as a refining step in SBPAX. Information about substances and processes extracted from databases (such as in BioPAX format) is presented as raw SBPAX which is converted into a model with user input. For example, if raw SBPAX describes an interaction of EGFR phosphorylation (multiple modifications of EGFR are involved) the way typically found in databases, we can not use database substance EGFR. Instead, we have to declare subset substances for unphosphorylated and phosphorylated EGFR respectively. The topology of the locations is usually not in the database, so it has to be added as well. Finally, a species (e.g. "EGFR in cytosol") is defined by a substance ("EGFR") and a location ("cytosol"). Refined SBPAX contains this inferred information.

subsets the final step would be to determine, which of these subsets participate in which interaction. After participating substances are determined, a user will be provided with an option to assign locations (if they are unavailable) and set topology of locations, such as set surroundedBy and hasDimension properties. Finally, the user will be provided with the list of processes. Some of the processes may be flagged, for example if previously the user identified reactant and product as the same species. The user will have an option to modify certain processes, which may lead to changes in substance assignment, etc.

After the selection of model elements, the next step is converting the SBPAX data into a fully annotated mathematical model, such as SBML. The SBML language specification has recently introduced features that facilitate and standardize the inclusion of additional information that is not required for the numerical interpretation of the model, but which can help describe the model and relate model and model elements to each other, both within the same file or between files from different sources.

5 VCell-Based Expert System for Building and Annotating Computational Models of Molecular Interaction Networks

The Virtual Cell (VCell; http://vcell.org/; [7], [8]) is a computational framework that is easily accessible to cell biologists and that permits construction of models, application of numerical solvers to perform simulations, and analysis of simulation results. Due to continuous enhancements in capabilities and to many unique features, it has achieved a fast growing user base. As of February, 2009, more then 2,000 worldwide VCell users had actually run simulations. These users are currently collectively storing over 29,000 models and the results of more than 160,000 simulations in the VCell database system, of which more than 600 models and more than 2,300 simulation results were made public by their owners to be available to the overall scientific community. Users can formulate complex models of cellular processes with a simple biology-oriented graphical user interface. VCell will automatically (i) generate the appropriate mathematical encoding for running a simulation, and (ii) generate and compile the appropriate computer code.

SyBiL was developed with VCell in mind. An expert system that enables and guides users to create, extend and annotate computational models from variety of web resources would be extremely useful for VCell framework. A typical computational model consists of compartments populated by substances interacting with each other. The expert system would support starting new models, as well as adding elements and annotations to existing models. The goal of the expert is to respond to any situation by presenting the user with a list of options that is short, but allows proceeding efficiently in any direction the user may desire.

Practically any situation can be boiled down to this scenario: the user needs to choose one specimen from a universe of particulars (e.g. a substance, an

interaction, an anatomical feature, an organism). The expert system activates a set of candidates and ranks them according to a variety of indicators of how likely it is the one the user intends and also places them into categories. Picking a candidate completes the selection process. If no single candidate can be selected, the expert may present a category of candidates, and picking a category leads to a new list of candidates.

Working in a context, an expert system will be able to make sophisticated guesses, for example when adding a new protein to a model containing human proteins, the new protein is probably human, too. Complex queries become possible, such as how do IGF1 and IL6 interact with each other?, which can only be answered by a complex series of queries.

6 Conclusions and Further Directions

We are developing the Systems Biology Linker (SyBiL) application and Virtual Cell (VCell) plugin as a tool that automatically retrieves public information while guiding the user through the process of modeling: for example, starting by choosing human as an organism and EGFR as a name, it will locate the UniProt identifier for human epidermal growth factor receptor, create a corresponding species element in a VCell model, and add standards-compliant annotations to identify human, human EGFR, related publications etc. It will offer many suggestions on how to proceed: what reactions are known to involve this protein, what are its possible subcellular locations, what are its most common binding partners, what reaction chains (signaling pathways) may lead to some desired effect (e.g. MAP kinase cascade activation), etc. All user decisions will be recorded, and at each point the user will be able to add extra information, with annotations being automatically updated. This will also ensure that modeling assumptions can easily retrieved in the future for model adjustments, and greatly facilitate model reuse in different contexts and/or by different researchers.

The primary challenge of this project is to keep up with an evolving web. Formats are being replaced by newer versions, and sources may change their interface or may go in and out of existence. To meet this challenge, we rely as much as possible on technology that is generic rather than specific to particular versions. For example, we use a generic RDF/OWL tool (Jena, [28]) to process BioPAX, and we base queries on a generic protocol. We also rely as much as possible on configuration rather than hard-coding. In particular, processing of BioPAX is based on a bridging ontology SBPAX, which makes the code itself independent of many details of BioPAX. Support for a new version of BioPAX is primarily added by extending SBPAX, and little or no change is necessary for the main application code.

Acknowledgements. The project was supported by NIH U54 RR022232 and P41RR013186 grants.

References

1. Vastrik, I., et al.: Reactome: a knowledgebase of biological pathways and processes. Genome Biol. 8(3) (2007); Database is accessible, http://www.reactome.org/
2. Krummenacker, M., Paley, S., Mueller, L., Yan, T., Karp, P.D.: Querying and computing with BioCyc databases. Bioinformatics 21(16), 3454–3455 (2005); Database is accessible at, http://biocyc.org
3. http://pid.nci.nih.gov/
4. Le Novere, N., et al.: BioModels Database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. Nucleic Acids Res. 34, D689–D691 (2006); Database is accessible at, http://biomodels.net
5. http://www.inoh.org/
6. Ogata, H., Goto, S., Sato, K., Fujibuchi, W., Bono, H., Kanehisa, M.: KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res. 27, 29–34 (1999); Database is accessible at, http://www.genome.jp/kegg/
7. Slepchenko, B.M., et al.: Quantitative cell biology with the Virtual Cell. Trends Cell Biol. 13, 570–576 (2003); Software is accessible at, http://vcell.org
8. Moraru, I.I., et al.: Virtual Cell modelling and simulation software environment. IET Systems Biology 2(5), 352–362 (2008)
9. Hoops, S., et al.: COPASI -a COmplex PAthway Simulator. Bioinformatics 22(24), 3067–3074 (2006), http://copasi.org
10. Funahashi, A., Tanimura, N., Morohashi, M., Kitano, H.: CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. BIOSILICO 1, 159–162 (2003); Software is accessible at, http://celldesigner.org
11. Luciano, J.S.: PAX of mind for pathway researchers. Drug Discov Today 10, 937–942 (2005)
12. Hucka, M., Finney, A., et al.: The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 19, 524–531 (2003)
13. Zinovyev, A., Viara, E., Calzone, L., Barillot, E.: BiNoM: a Cytoscape plugin for manipulating and analyzing biological networks. Bioinformatics 24, 876–877 (2008)
14. Shannon, P., et al.: Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Res. 13(11), 2498–2504 (2003)
15. Demir, E., Babur, O., Dogrusoz, U., Gursoy, A., Nisanci, G., Cetin-Atalay, R., Ozturk, M.: PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways. Bioinformatics 18(7), 996–1003 (2002)
16. Ruebenacker, O., Moraru, I.I., Schaff, J.C., Blinov, M.L.: Kinetic Modeling Using BioPAX Ontology. In: Proceedings of the 2007 IEEE International Conference on Bioinformatics and Biomedicine, pp. 339–348 (2007)
17. Blinov, M.L., Ruebenacker, O., Moraru, I.I.: Complexity and modularity of intracellular networks: a systematic approach for modelling and simulation. IET Systems Biology 2(5), 363–368 (2008)
18. Pathway Commons, http://www.pathwaycommons.org/pc/
19. Belleau, F., Nolin, M.A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: towards a mashup to build bioinformatics knowledge systems. J. Biomed Inform. 41(5), 706–716 (2008), http://bio2rdf.org/
20. Universal Protein Resource, http://uniprot.org
21. Blinov, M.L., Faeder, J.R., Goldstein, B., Hlavacek, W.S.: A network model of early events in epidermal growth factor receptor signaling that accounts for combinatorial complexity. Biosystems 83(2-3), 136–151 (2006)

22. Ruebenacker, O., Moraru, I.I., Schaff, J.C., Blinov, M.L.: Integrating BioPAX knowledge with SBML models. IET Systems Biology 3(5), 317–328 (2009)
23. Le Novére, N., Finney, A., Hucka, M., et al.: Minimum information requested in the annotation of biochemical models (MIRIAM). Nature Biotechnology 23(12), 1509–1515 (2005)
24. Le Novere, N.: Model storage, exchange and integration. BMC Neurosci. 7 (suppl. 1), S11 (2006)
25. Ashburner, M., et al.: Gene Ontology: tool for the unification of biology. Nat. Genet. 25, 25–29 (2000)
26. Mayer, B.J., Blinov, M.L., Loew, L.M.: Molecular Machines or Pleiomorphic Ensembles: Signaling Complexes Revisited. J. Biol. 8(9), 81–95 (2009)
27. OWL Web Ontology Language, http://www.w3.org/TR/owl-features/
28. Jena Semantic Web Framework, http://jena.sourceforge.net

Touring Protein Space with Matt

Noah Daniels, Anoop Kumar, Lenore Cowen*, and Matt Menke

Department of Computer Science, Tufts University
161 College Ave, Medford, MA 02155
cowen@cs.tufts.edu

Abstract. Using the Matt structure alignment program, we take a tour of protein space, producing a hierarchical clustering scheme that divides protein structural domains into clusters based on geometric dissimilarity. While it was known that purely structural, geometric, distance-based metrics of structural similarity, such as Dali/FSSP, could largely replicate hand-curated schemes such as SCOP at the family level, it was an open question as to whether any such scheme could approximate SCOP at the more distant superfamily and fold levels. We partially answer this question in the affirmative, by designing a clustering scheme based on Matt that approximately matches SCOP at the superfamily level. Implications for the debate over the organization of protein fold space are discussed.

1 Introduction

The accepted gold-standard hierarchical classification systems for protein structural domains, SCOP [21,2] and CATH [22,23,11], have long relied on manual classification methods to organize the hierarchy and place new protein structures within their framework. Even now, where both SCOP and CATH have switched to hybrid manual/semi-automated methods [11], the automatic methods are still attempting to fit new protein domain folds into an initial classification schema that was derived manually. New modifications to the clustering structure continue to be made by expert biologists based on sequence, evolutionary, and functional information, not solely based on geometric similarity of the placement of atoms in the protein backbone.

On the other hand, pairwise protein structural alignment programs superimpose protein domains to minimize a distance metric based solely on geometric criteria [8]. When such a scheme is coupled with one of many possible methods that create hierarchical clusters based on pairwise distance metrics [29], the result is a fully automatic, unsupervised partitioning of protein structural domains into hierarchical classification systems. Such "bottom up" protein structure classifications, as they are called in Valas et al. [33], have been previously designed based on VAST [19,10], Dali [16,17,15] and others [36], and have both practical and theoretical appeal. Practically, removing a human expert speeds the assignment of new protein structures to clusters. Theoretically, a mathematical characterization of protein similarity and dissimilarity, if it proves biologically useful or meaningful, is objective, uniformly applied, and gives a human-expert-independent map of the known protein universe.

* Corresponding author.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 18–28, 2010.

Unfortunately, it has been found in multiple previous papers that SCOP and CATH hierarchical classifications of protein structure both differ substantially from each other [12,9,7] and also from the classification schema that result from automatic bottom-up unsupervised clusterings of protein space [8,12,30,7,28], even when protein chains are broken up into the more modular unit of "protein domain," as is now done by SCOP, CATH, and most automated schemes [17,33]. Previous papers have characterized those protein domain clusters on which SCOP and CATH agree [12,9,7]. Previous automatic methods seem to be able to be made to match the closest-homology *family* level of the SCOP hierarchy, but were found to diverge considerably at the more distantly homologous *superfamily* and at the *fold* levels of the SCOP hierarchy [8,12,30,18,28,32], with similar divergence from CATH [12,13,7] This is unfortunate, because, for example, the superfamily level of the SCOP hierarchy clusters proteins that share similar folds and are believed to have evolved from a common ancestor [21], allowing important inferences to be made about function [28,33]. Thus the superfamily level of the SCOP hierarchy has strong biological utility (we focus on SCOP rather than CATH for the remainder of this paper; similar statements can be made about CATH): if a fully automated "bottom-up" distance-based clustering methods cannot approximately replicate it, it is not clearly meaningful or useful.

This ties into a spirited debate among the computational proteins community, about the central question of whether "protein fold space" is *discrete* or *continuous* [26]. A continuous view comes from the theory that modern protein evolved by aggregating fragments of ancient proteins [26,13,33,27]. A discrete view comes from evolutionary process constrained by thermodynamic stability of the structure [27]. In particular, if most mutations move the confirmation of a stable folded chain away from an "island" of thermodynamic structural stability, then stabilizing selection will promote fold conservation, and movements between folds will be uncommon [6]. If geometric distance and evolutionary relation approximately coincide, then an automatic method that approximately matches SCOP at the superfamily level is conceivable.

In this paper, we present a bottom-up automatic hierarchical classification scheme for protein structural domains based on the multiple structure alignment program Matt [20]. Matt, which stands for "multiple alignment with translations and twists" was specifically developed by our group to geometrically align more distantly homologous protein domains. It accomplishes this by allowing flexibility in the form of small, geometrically impossible bends and breaks in a protein structure, in order to distort it into alignment with another protein structure. Matt was shown to perform particularly well compared to competing multiple and pairwise structure alignment programs on proteins whose homology was similar to the SCOP superfamily level [20,25,3]. Surprisingly, we find that our automatic classification scheme based on a pairwise distance metric derived from Matt, coupled with a straightforward neighbor-joining algorithm to construct the hierarchical clusters [31] matches SCOP better than previous automatic methods, *at the superfamily, and even, to some extent, at the fold level.* In comparison, the same hierarchical clustering method using a pairwise distance metric based on DaliLite [15], a recent implementation of the Dali structural alignment program, replicates previous findings and cannot mimic SCOP on the superfamily level of the SCOP hierarchy. We thus conclude that perhaps protein domain space is naturally discrete (at least through

the superfamily level), and that perhaps the manually curated SCOP hierarchy has *geometric* coherence at the superfamily level (and in some parts of the fold hierarchy, see Discussion) so these clusters are intrinsic properties of the geometry of fold space, not just human-generated categories.

A practical implication of our results may be that automatic methods with a Matt-based distance metric may ultimately help speed the assignment of new protein structural domains to the appropriate place in the SCOP hierarchy. We note, however, that in fact determining where to place a new structure into an existing hierarchy is a much simpler problem (analogous to "supervised learning") than creating an entire cluster hierarchy from an automatic pairwise distance metric from scratch (analogous to "unsupervised learning"), and fairly successful methods already exist to correctly place a new structure into the existing SCOP hierarchy [9,4,5]. Thus the primary interest in this result may be that if a Matt distance metric can "recover" SCOP superfamilies to a great extent, this validates both automatic and hand-curated methods of classification, and the entire concept of "superfamily" at the same time. Namely, at this level of structural similarity, it appears we may not often have to choose between evolutionary and geometric criteria for structural domain similarity.

Finally, we remark that this work, like most recent work that compares different hierarchical classification systems, already presumes the "structural domain" as the basic structural unit (as do SCOP and CATH), where many protein structures contain multiple structural domains [17]. The problem of partitioning a protein into its structural domains is far from trivial [34,14] but there has been much recent progress in computational methods that split a protein structure automatically into domains and find the domain boundaries [14,24]. In any case, that is not the focus of our current paper, and we assume the protein has already been correctly split into domains as a preprocessing step.

2 Methods

2.1 Representative Proteins

From the 110,776 protein domains of known structure from ASTRAL version 1.75, we construct a set of representative protein domains filtered to 80% identity (according to BLASTP [1]) and a minimum sequence length of 40 residues. This provides a reasonable first pass for identifying groups of similar protein domains, and allows us to shrink the search space significantly. The set of clusters was constructed by running a greedy agglomerative minimum-linkage clustering algorithm based on this threshold of 80% sequence identity. This produced 10,418 groups of proteins that shared significant sequence identity.

From each cluster, we identified a representative. First, we preferred non-engineered, non-mutation proteins having an X-ray crystallography resolution of ≤ 5.0 Angstroms. Next, treating each cluster as a (potentially, but not necessarily complete) graph whose nodes are the constituent proteins and whose edge weights are the sequence identity values from the BLASTP alignments with at least 80% identity, we consider the weighted degree (sum of edge weights) of each protein, and we favor the proteins with greatest weighted degree. We break ties first by the date the structure was determined (preferring

more recent structures), then by the quality of the solved structure. The remaining ties typically come from sequences with $\geq 99\%$ identity, and we break them arbitrarily. The resulting set has 10,418 representative protein domains.

2.2 Distance Metrics

For these 10,418 representatives, we performed an all-pairs structural alignment using both DaliLite [15], the structural aligner used in the FSSP classification scheme [17] and Matt. In each case, a distance (or dissimilarity) metric is derived for each pair. For DaliLite, the Z-score proved to be a good metric, so we used it without further modification.

For Matt, we used a new distance metric that is a modification of the p-value metric computed in [20]. Let c be the length of the aligned core shared between the two proteins (in residues), r be the RMSD (root mean square deviation) of the alignment, l_1 and l_2 be the lengths of the two protein domains being aligned (in residues), and k_1, k_2, and k_3 be the constants from the Matt p-value. We compute the distance between two Matt-aligned proteins as follows:

$$d = \frac{1}{k_1 \times (r - k_2 \times \frac{c^2}{\frac{l_1+l_2}{2}} + k_3)}$$

This metric differs from the formula that Matt uses to compute a p-value only in that it squares the core-length term to better weight longer aligned cores (c^2 instead of c). We found this improved performance.

2.3 Distance Threshold

Based on each of the Dali Z-score and Matt distance metrics, we next learned the distance cutoffs that most closely mimicked the family, superfamily, and fold levels of the SCOP hierarchy as follows:

1. Initialize a training set T and a set of already-chosen pairs A
2. 10,000 times, do:
   (a) Choose proteins p and q such that $p \neq q$ and p and q are in the same SCOP grouping, and the pair $p, q \notin A$
   (b) Choose proteins r and s such that $r \neq s$ and r and s are in different SCOP groupings, and the pair $r, s \notin A$
   (c) Add p, q and r, s to A
   (d) Determine the DaliLite or Matt distance between p and q. Call this $d_{p,q}$
   (e) Add $d_{p,q}$ to the training set T with label *true*
   (f) Determine the DaliLite or Matt distance between r and s. Call this $d_{r,s}$
   (g) Add $d_{r,s}$ to the training set T with label *false*
3. Compute true positive rate R_{tp}, true negative rate R_{tn}, positive rate R_p, and negative rate R_n for T based on the class labels $true$ and $false$
4. Determine the value of $d_{p,q}$ that results in maximizing the accuracy $\frac{R_{tp}+R_{tn}}{R_p+R_n}$

In other words, we set $d_{p,q}$ to be the value corresponding to the point on the Receiver Operating Characteristic (ROC) curve that intersects the tangent iso-performance line [35], or maximizing the sum $R_{tp} + R_{tn}$. The area under the ROC curve measure

(AUC) is a summary statistic that captures how well the pairwise distance metric can discriminate between structures that share or do not share SCOP cluster membership.

We note that setting the pairwise distance cutoffs (determining the value of $d_{p,q}$ in step 4) is the only "supervision" our algorithm uses in constructing its clustering (see discussion below). *Once the pairwise distance cutoff is set, no further information from SCOP is utilized to produce the clustering.*

2.4 Clustering and Tree-Cutting

Based on this distance function, we computed values for all pairwise alignments based on the Matt or DaliLite output, and represented this as a distance matrix. We ran the ClearCut program [31] in strict neighbor-joining mode (-N option) to produce a dendrogram based on these Matt or DaliLite distance values. We then recursively descended this tree to produce family, superfamily, and class-level groupings as follows. For a given subtree, if all leaves (protein domains) in that subtree are within a threshold t of one another (where t is the family, superfamily, or fold threshold), then those leaves are all merged into a new grouping of that level. Otherwise, we recursively descend into the two subtrees of that subtree's root until we reach a subtree all of whose leaves fall within a given threshold (family, superfamily, or fold; based on Matt distance or DaliLite Z-score as appropriate) of one another. Thus, we are performing a total-linkage clustering, but using the topology of the dendrogram to determine which protein domains get left out of a given cluster.

We remark that Sam, et al. [29] did an extensive study of clustering and tree-cutting methods, and looked at their effect on performance for several distance metrics. They tested 3 "SCOP-dependent" and 7 "SCOP-independent" tree-cutting strategies. However, their "SCOP-independent" strategies all required as input the target number of SCOP clusters to produce at each level. In contrast, our method discovers the number of clusters as an organic function of the protein domain space; it is thus of independent interest that we nearly replicate the number of SCOP clusters at each level (see Table 2).

2.5 Jaccard Similarity Metric

The Jaccard index, or Jaccard similarity coefficient, of two sets A and B is defined as $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$. Based on the Jaccard index of a cluster (e.g. family or superfamily or fold) produced by our algorithm (a Matt family or DaliLite family) and a SCOP grouping of the same level, and looking at the identity of protein domains in the two groupings, we can compare how alike they are. We can thus easily find the most similar SCOP family to each Matt family, $S \rightarrow M$ and vice versa, $M \rightarrow S$. This directional mapping is neither one-to-one nor onto, but each cluster on the 'source side will be mapped to some most-similar cluster on the 'sink side. The resulting directed graph allows us to explore the distribution of Jaccard indices as well as the distribution of degrees of each cluster. A perfect matching would correspond to every Jaccard index being 1.0, and every cluster having degree 1. Clearly, we do not expect to achieve a perfect matching but this metric allows us to compare the quality of clustering, relative to SCOP, of our algorithm using the Matt distance metric and the DaliLite Z-score distance metric.

Each direction of the metric is produced as follows, using as an example the comparison of Matt families to SCOP families. Consider the set of Matt families and SCOP families as a bipartite graph, with the Matt families on one side of the bipartition and the SCOP families on the other. Initially, the graph has no edges. For each Matt family, find the most similar (by Jaccard index) SCOP family. A weighted, directed edge is drawn from each Matt family to its most similar SCOP family; the edge weight is equal to the Jaccard index, which ranges from 0 to 1. This is performed until each Matt family has been matched to a SCOP family. This process is repeated in the other direction, matching each SCOP family to its most similar Matt family, and the same thing is done for Matt and DaliLite at the superfamily and fold levels of the SCOP hierarchy.

Recall that in this analysis, as is standard [12], we are considering only the protein domains that were identified as cluster representatives within each group of protein domains that share 80% sequence identity.

3 Results

3.1 Pairwise Distance Comparisons

Table 1 shows the AUC at the SCOP family, superfamily, and fold level, for the Matt and DaliLite distance metrics. Note that at the family and fold levels, these values are very close (DaliLite outperforms Matt by a small margin), but at the superfamily level, Matt significantly outperforms DaliLite, achieving 0.842 ROC Area vs. DaliLite's 0.615. Matt was developed to better align structures at the superfamily level of homology, but the size of the gap in ROC AUC is still surprising.

Table 1. ROC Area for pairwise performance vs. SCOP. While DaliLite slightly outperforms Matt at family and fold levels, Matt significantly outperforms DaliLite at the superfamily level.

	Matt	DaliLite
Families	0.922	0.958
Superfamilies	0.842	0.615
Folds	0.840	0.871

3.2 Clustering Performance

While the pairwise performance of Matt compared to DaliLite is impressive, pairwise similarity does not necessarily translate into better clustering performance. Thus it is Matt's clustering performance we explore next. First we give the simplest possible comparison; raw numbers of clusters produced by Matt and DaliLite compared to SCOP at the three levels. Recall that unlike the clustering algorithm explored by [29], the number of clusters produced by our dendrogram and tree-cutting method is a direct consequence of the pairwise distance metric threshold, and is not artificially set to match SCOP (see section 2.4). Table 2 shows that the Matt clustering produces approximately the same number of clusters as SCOP at all three levels. While DaliLite also produces approximately the same number of clusters at the family level, at the superfamily and fold levels it produces many more clusters than SCOP. We explore exactly how both methods split and merge SCOP clusters in more detail next.

Table 2. Number of clusters at each level for each method. Matt more closely matches the number of families, superfamilies, and folds in SCOP than DaliLite does. DaliLite clustering results in too few families, but too many superfamilies and folds with respect to SCOP.

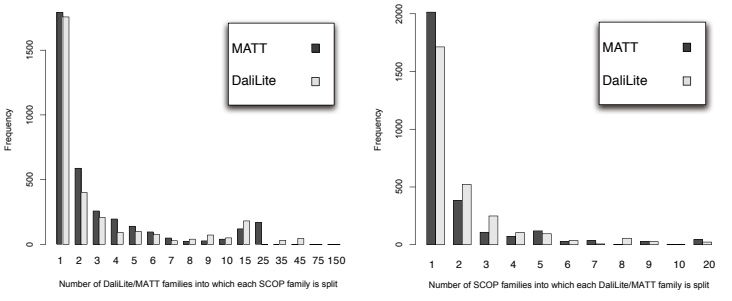
	SCOP	Matt	DaliLite
Families	3471	3498	3081
Superfamilies	1656	1716	2455
Folds	981	891	2277

Table 3. Descriptive statistics for the family, superfamily, and fold levels of classification. μ Degree is the average number of clusters from the first scheme that map to a single cluster in the second, and σ Degree gives the standard deviation. Similarly, we give min, μ, and σ of the Jaccard similarity.

Family	Max Deg.	μ Deg.	σ Deg.	Min Sim.	μ Sim.	σ Sim.
Matt →SCOP	30	3.63	5.470	0.005	0.611	0.373
DaliLite → SCOP	45	3.902	6.919	0.001	0.598	0.380
SCOP → Matt	15	1.873	2.160	0.127	0.712	0.336
SCOP → DaliLite	12	1.983	1.823	0.001	0.655	0.347
Superfamily	Max Deg.	μ Deg.	σ Deg.	Min Sim.	μ Sim.	σ Sim.
Matt → SCOP	28	3.633	5.094	0.003	0.587	0.389
DaliLite → SCOP	153	16.61	36.54	0.001	0.428	0.406
SCOP → Matt	15	1.704	1.913	0.020	0.714	0.326
SCOP → DaliLite	10	1.470	1.229	0.001	0.713	0.324
Fold	Max Deg.	μ Deg.	σ Deg.	Min Sim.	μ Sim.	σ Sim.
Matt → SCOP	18	3.719	4.258	0.004	0.467	0.354
DaliLite → SCOP	149	26.57	40.87	0.001	0.321	0.389
SCOP → Matt	6	1.958	1.122	0.022	0.512	0.326
SCOP → DaliLite	3	1.117	0.353	0.001	0.758	0.299

The Jaccard index serves as a good indicator of how well Matt and DaliLite match SCOP. As the raw numbers of clusters in table 2 suggest, DaliLite often shatters SCOP superfamilies into multiple clusters. Interestingly, DaliLite also shatters SCOP folds into many more shards on average than Matt. How can this be given the very similar pairwise classification performance at the fold level? We defer this question until the discussion section. We note that even at the family level, Matt performs slightly better than DaliLite at both the average degree and average Jaccard similarity metrics. The average number of Matt or DaliLite families that match to a single SCOP family is between 3.5 and 4; however, notice that a large majority of Matt or DaliLite families map to a single SCOP family and the average is pulled up by a few outliers (see histograms in figure 2). Average degree values at the superfamily and fold levels stay nearly constant for Matt, whereas DaliLite's average degree values rise to 16.61 for the superfamily level and 26.57 at the fold level. In the other direction, considering how many Matt or

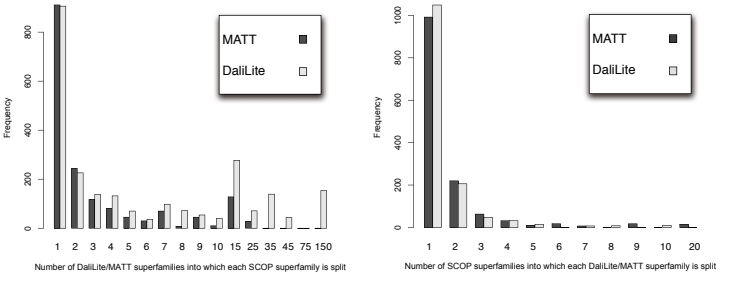
DaliLite clusters span multiple SCOP clusters, at the family level the average degree for Matt and DaliLite are nearly identical (between 1.8 and 2). At the superfamily and fold levels, we would expect DaliLite to outperform Matt by virtue of the fact that it creates many smaller clusters (see table 2), and DaliLite does, but by a fairly small margin (1.4 to 1.7 at the superfamily level and 1.1 to 2 at the fold level). The distributions are displayed in more detail in the histograms in figures 1, 2, and 3.



(a) Number of Matt vs. DaliLite families into which each SCOP family is shattered

(b) Number of SCOP families into which each Matt or DaliLite family is shattered

Fig. 1. Family level splitting behavior



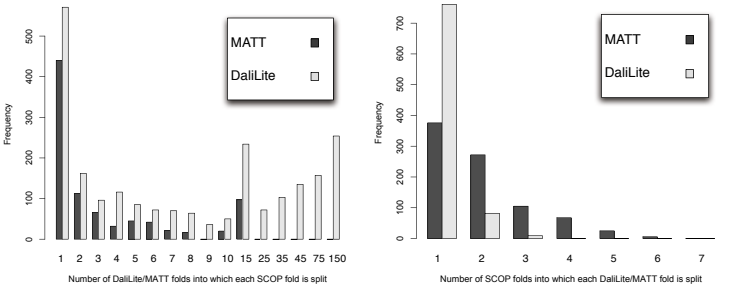
(a) Number of Matt vs. DaliLite superfamilies into which each SCOP superfamily is shattered

(b) Number of SCOP superfamilies into which each Matt or DaliLite superfamily is shattered

Fig. 2. Superfamily level splitting behavior

4 Discussion

We have shown that using more modern structure alignment programs, an automatic clustering method that approximates SCOP at a superfamily level may be feasible. Of course, any mapping between clusters based on geometric equivalence, and clusters seeking to capture evolutionary and geometric equivalence using information beyond geometry will be imperfect — yet the Matt clusters at the superfamily level seem sufficiently interesting that differences between Matt and SCOP could be illuminating.



(a) Number of Matt vs. DaliLite folds into which each SCOP fold is shattered

(b) Number of SCOP folds into which each Matt or DaliLite fold is shattered

Fig. 3. Fold level splitting behavior

As noted earlier, DaliLite tends to shatter SCOP folds into many more shards than Matt. How can this be given the very similar pairwise classification performance at this level? One possibility is that the Matt-based distance metric is more stable in regions far beyond the specific thresholds we learned, and that this leads to the topology of the resulting dendrogram (before cutting) more faithfully representing the relationships between more and less closely related folds. In other words, DaliLite's Z-scores may result in more 'spoilers' that break up clusters (due to our total-linkage requirement) than Matt's distance metric.

An interesting question is what Matt clustering results mean for protein fold space at the "fold" level of structural homology. Here, while the Matt clustering clearly seems more informative than that produced by DaliLite, performance is still uneven. There seem to be some SCOP folds where the Matt split appears meaningful, and others where it is more arbitrary. For example, a notoriously difficult SCOP fold for multiple automatic methods is the enormous β/α TIM barrel fold. SCOP places 33 separate superfamilies into this one fold, but both of our clustering approaches seem to split this into multiple folds. For example, DaliLite splits the TIM barrel SCOP fold into 106 separate folds. Matt splits the TIM barrel SCOP fold into 'only' 17 separate folds, which is better than 106, but inspection of the boundaries between these Matt fold classes shows more continuity of shape, and the cuts appear to be somewhat arbitrary.

Thus, while touring protein space with Matt seems to lend support to a more discrete view of protein space through the superfamily level, further study of individual clusters may be warranted to determine the breakpoint distance at which continuity takes over. Perhaps the degree of similarity of different individual SCOP folds can be characterized, similarly to what Suhrer, et al. [32] did at the family level.

Acknowledgements

This work was funded in part by NIH grant 1R01GM080330-01A1 (to LC).

References

1. Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., Lipman, L.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25, 3389–3402 (1997)
2. Andreeva, A., Howorth, D., Brenner, S., Hubbard, T., Chothia, C., Murzin, A.: SCOP database in 2004: refinements integrate structure and sequence family data. Nucleic Acids Research 32, D226–D229 (2004)
3. Berbalk, C., Schwaiger, C., Lackner, P.: Accuracy analysis of multiple structure alignments. Protein Science 18, 2027–2035 (2009)
4. Cheek, S., Qi, Y., Krishna, S., Kinch, L., Grishin, N.V.: SCOPmap: Automated assignment of protein structures to evolutionary superfamilies. BMC Bioinformatics 7 (2006)
5. Chi, P.-H., Shyu, C.-R., Xu, D.: A fast SCOP fold classification system using content-based E-predict algorithm. BMC Bioinformatics 7, 10.1186/1471–2105–7–362 (2006)
6. Choi, I.-G., Kim, S.-H.: Evolution of protein structural classes and protein sequence families. Proc. Nat. Acad. Sci. 103, 14056–14061 (2006)
7. Day, R., Beck, D., Armen, R., Daggett, V.: A consensus view of fold space: Combining SCOP, CATH, and the Dali domain dictionary. Protein Science 12, 2150–2160 (2003)
8. Gerstein, M., Levitt, M.: Comprehensive assement of automatic structural alignment against a manual standard, the SCOP classification of proteins. Protein Sci., 445–456 (1998)
9. Getz, G., Vendruscolo, M., Sachs, D., Domany, E.: Automatic assignment of SCOP and CATH protein structure classifications from FSSP scores. Proteins: Structure Function and Genetics 46, 405–415 (2002)
10. Gibrat, J., Madej, T., Bryant, S.: Suprising similarities in structure comparison. Curr. Opin. Struct. Biol. 6, 377–385 (2006)
11. Greene, L., Lewis, T., Addou, S., Cuff, A., Dallman, T., Dibley, M., Redfern, O., Pearl, F., Nambudiry, R., Reid, A., Silitoe, I., Yeats, C., Thornton, J., Orengo, C.: The CATH domain structure database: new protocols and classification levels give a more comprehensive resource for exploring evolution. Nucleic Acids Res. 35, D291–D297 (2007)
12. Hadley, C., Jones, D.: A systematic comparison of protein structure classifications: SCOP, CATH and FSSP. Structure 7, 1099–1112 (1999)
13. Harrison, A., Pearl, F., Mott, R., Thornton, J., Orengo, C.: Quantifying the similarity within fold space. J. Mol. Bio. 323, 909–926 (2002)
14. Holland, T., Veretnik, S., Shindyalov, I.N., Bourne, P.: Partitioning protein structures into domains: why is it so difficult? J. Mol. Biol. 361, 562–590 (2006)
15. Holm, L., Park, J.: DaliLite workbench for protein structure comparison. Bioinformatics 16, 566–567 (2000)
16. Holm, L., Sander, C.: Mapping the protein universe. Science 260, 595–602 (1996)
17. Holm, L., Sander, C.: Touring protein fold space with Dali/FSSP. Nucleic Acids Res., 316–319 (1998)
18. Kolodny, R., Petrey, D., Honig, B.: Protein structure comparison: implications for the nature of fold space, and structure and function prediction. Curr. Opin. Struct. Biol. 16, 393–398 (2006)
19. Madej, T., Gibrat, J.-F., Bryant, S.: Threading a database of protein cores. Proteins 23, 356–369 (1995)
20. Menke, M., Berger, B., Cowen, L.: Matt: Local flexibility aids protein multiple structure alignment. PLoS Comput. Biol. 4(1), e10 (2008) doi:10.1371/journal.pcbi.0040010
21. Murzin, A., Brenner, S., Hubbard, T., Chothia, C.: SCOP: a structural classification of proteins database for the investigation of sequences and structures. J. Mol. Biol. 297, 536–540 (1995)

22. Orengo, C., Michie, A., Jones, S., Jones, D., Swindells, M., Thornton, J.: Cath- a hierarchic classification of protein domain structures. Structure 5(8), 1093–1108 (1997)
23. Pearl, F., Bennett, C., Bray, J., Harrison, A., Martin, N., Shepherd, A., Sillitoe, I., Thornton, J., Orengo, C.: The CATH database: an extended protein family resource for structural and functional genomics. Nucleic Acids Res. 31, 452–455 (2003)
24. Redfern, O., Harrison, A., Dallman, T., Pearl, F., Orengo, C.: CATHEDRAL: A fast and effective algorithm to predict folds and domain boundaries from multidomain protein structures. PLOS Computational Biology, e232 (2007) doi:10.1371/journal.pcji.0030232
25. Rocha, J., Segura, J., Wilson, R., Dasgupta, S.: Flexible structural protein alignment by a sequence of local transformations. Bioinformatics 25, 1625–1631 (2009)
26. Rost, B.: Did evolution leap to create the protein universe? Curr. Opinion in Struct. Biol., 409–416 (2002)
27. Sadreyev, R., Kim, B.-H., Grishin, N.: Discrete-continous duality of protein structure space. Curr. Opinion Structural Biol. 19, 321–328 (2009)
28. Sam, V., Tai, C., Garnier, J., Gibrat, J.F., Lee, B., Munson, P.: ROC and confusion analysis of structure comparison methods identify the main causes of divergence from manual protein classification. BMC Bioinformatics 7, 206 (2006)
29. Sam, V., Tai, C., Garnier, J., Gibrat, J.F., Lee, B., Munson, P.: Towards an automatic classification of protein structural domains based on structural similarity. BMC Bioinformatics 9 (2008)
30. Shindyalov, I., Bourne, P.: An alternative view of protein fold space. Proteins 38, 513–514 (2000)
31. Simonsen, M., Mailund, T., Pedersen, C.N.S.: Rapid neighbour-joining. In: Crandall, K.A., Lagergren, J. (eds.) WABI 2008. LNCS (LNBI), vol. 5251, pp. 113–122. Springer, Heidelberg (2008)
32. Suhrer, S., Wederstein, M., Sippl, M.: QSCOP-SCOP quantified by structural relationships. Bioinformatics 23, 513–514 (2007)
33. Valas, R., Yang, S., Bourne, P.: Nothing about protein structure classification makes sense except in the light of evolution. Curr. Opin. Struct. Biol. 19, 329–334 (2009)
34. Veretnik, S., Bourne, P., Alexandrov, N., Shindyalov, I.: Toward consistent assignment of structural domains in proteins. J. Mol. Biol. 339, 647–678 (2004)
35. Vuk, M., Curk, T.: Roc curve, lift chart and calibration plot. Metodolo ski zvezki 2, 89–108 (2006)
36. Zemla, A., Geisbrecht, B., Smith, J., Lam, M., Kirkpatrick, B., Wagner, M., Slezak, T., Zhou, C.: STRALCP–structure alignment-based clustering of proteins. Nucleic Acids Res. 35, e150 (2007)

Fixed-Parameter Algorithm for General Pedigrees with a Single Pair of Sites

Duong D. Doan and Patricia A. Evans

Faculty of Computer Science
University of New Brunswick, Fredericton, New Brunswick, Canada
{b89ct,pevans}@unb.ca

Abstract. The problem of computing the minimum number of recombination events for general pedigrees with two sites for all members is investigated. We show that this NP-hard problem can be parametrically reduced to the Bipartization by Edge Removal problem and therefore can be solved by an $O(2^k \cdot n^2)$ exact algorithm, where n is the number of members and k is the number of recombination events.

Keywords: Fixed-parameter algorithm, haplotyping, pedigree.

1 Introduction

Human genomes contain two copies of each chromosome. Research shows that single chromosomes, called haplotypes, are useful to study complex genetic diseases [5]. While genomic data, called genotypes, are abundant and easy to collect, haplotypes are rare and much more difficult to obtain by a biochemical method. Therefore, a computational method to infer haplotypes from genotype data, called haplotyping, is necessary. Genotypes can be obtained from a population group where relationships between members are unknown or from a multigenerational family pedigree with known relationships between members. We only consider pedigree data in this paper.

In the absence of recombination events, haplotypes of members in a pedigree follow the Mendelian law of inheritance, where the two haplotypes of a child are transferred from its parents, one haplotype from its father and the other from its mother. Various haplotyping algorithms exist for non-recombinant pedigree data [1] [2] [15] [17], especially a linear algorithm for non-recombinant tree pedigrees [1] and a near-linear algorithm for non-recombinant general pedigrees [2]. Haplotype inference is complicated by recombination events and the complex structures of the data themselves. Recombination happens when complementary parts of both of a parent's haplotypes can be inherited as a single combined haplotype of a child (Figure 1). Structures of the pedigree data can be complex with loops, where there are multiple inheritance paths between some family members.

The haplotyping problem has been studied extensively in the last few years, both for pedigree and population data. If recombinations are allowed, the problem of inferring haplotypes for pedigrees with the minimum number of recombinations is NP-hard [8]. In fact, inferring haplotypes for pedigrees with minimum

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 29–37, 2010.

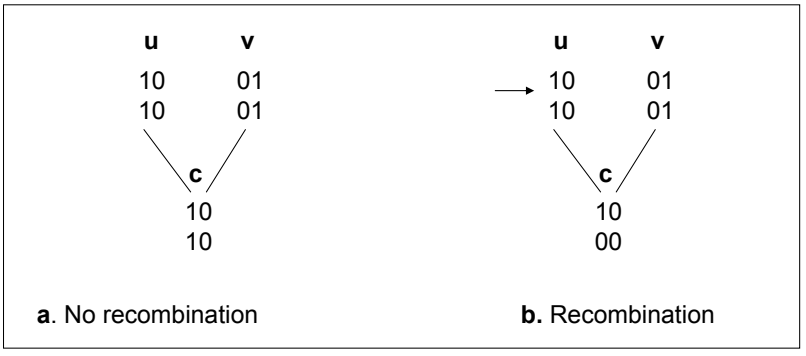


Fig. 1. Non-recombination vs. recombination. Recombination happens between sites 1 and 2 of parent u and the child c receives a combined haplotype from parent u. Here haplotypes of members are displayed in columns.

number of recombinations is NP-hard even for general pedigrees with only two sites or tree pedigrees with multiple sites [10]. For reconstructing haplotype configurations for pedigree data, Qian and Beckmann [12] proposed a rule-based algorithm with a time complexity $O(2^d n^2 m^3)$, where d is the largest number of children in a family, n is the number of members and m is the number of sites. The main principle of their algorithm is that the best haplotype configuration for pedigree data is the one that minimizes the number of recombination events (the *Minimum-Recombinant Haplotype Configuration (MRHC) problem*). In [7] [8] Li and Jiang proposed an $O(dmn)$ block-extension algorithm for the MRHC problem using a greedy heuristic to resolve adjacent sites. However, as discussed in [9], this algorithm did not always find the haplotypes that minimized the number of recombinations, and worked under some restrictions. In order to improve the performance and handle missing data, an integer linear programming (ILP) formulation [9] was proposed, in which a branch-and-bound algorithm was used to narrow the search space.

We study the minimum haplotype configuration for general pedigrees, where each member in a pedigree has only two sites; even this restricted problem is NP-hard [8]. We assume that there are no data missing and no data errors from the input genomic data. We prove that our problem can be reduced to the problem of finding the *line index* of a *signed graph* [16]. We further show that finding the line index of a signed graph can also be reduced to the Bipartization by Edge Removal problem. Our problem can therefore be solved by a fixed-parameter algorithm with a running time of $O(2^k \cdot n^2)$, where n is the number of members and k is the number of recombination events.

2 Concepts

A member is an individual. A set of members is called a *family* if it includes only two parents and their children; it is a *parent-offspring trio* (hereafter a *trio*)

if only two parents and one child are considered. A set of families connected through known family relationships is a *pedigree*. A parent is an *internal parent* if it is a child of another family; it is an *external parent* otherwise.

In diploid organisms, a cell contains two copies of each chromosome. The description data of the two copies are called a *genotype* while those of a single copy are called a *haplotype*. A specific location in a chromosome is called a *site* and its state is called an *allele*. There are two main types of sites, *microsatellites* and *single nucleotide polymorphisms*. A microsatellite site has several different states while a single nucleotide polymorphism *(SNP)* site has exactly two possible states, denoted by 0 and 1. Only SNPs are considered in this paper, as in other works on haplotype inference.

If the states at a specific site in two haplotypes are the same, then this site is a *homozygous* site (0-0 or 1-1); if they differ, it is *heterozygous* (0-1 or 1-0). Two haplotypes combine together to form one genotype. Each member u has two haplotypes, denoted by $h1_u$ and $h2_u$, which are vectors of 0 and 1's of length m, where m is the number of sites. The genotype of u, g_u, is a vector of 0's, 1's and 2's of length m, where $g_u[i] = 0$ means $h1_u[i] = 0 = h2_u[i]$, $g_u[i] = 1$ means $h1_u[i] = 1 = h2_u[i]$, and where $g_u[i] = 2$ means $\{h1_u[i], h2_u[i]\} = \{0, 1\}$. We say $h1_u$ and $h2_u$ are consistent with g_u. The complement haplotype of a haplotype h at a heterozygous site is denoted by $\bar{h}$, where $\bar{h} = 1 - h$ so, $\bar{0} = 1$ and $\bar{1} = 0$.

The problem in this paper is to find the haplotypes $h1_u$ and $h2_u$ for all members u that minimize the number of recombination events, given their genotypes g_u. A set of haplotypes found for all members is called a *haplotype configuration*. When $g_u[i] = 0$ or 1, then $h1_u[i]$ and $h2_u[i]$ are known, but if $g_u[i] = 2$, we may not yet know the value of $h1_u[i]$ and $h2_u[i]$, in which case we give them the value "?", and say that the site is *unresolved*. Our problem is defined as follows.

2-site-MRHC$_{opt}$: *Given the genotypes of a general pedigree P containing n members, where each member has only two sites, find a haplotype configuration that minimizes the number of recombination events.*

This optimization problem, called 2-site-MRHC$_{opt}$, was proven NP-hard [10]. We investigate the corresponding decision version of 2-site-MRHC$_{opt}$.

2-site-MRHC$_k$: *Given a positive integer k and the genotypes of a general pedigree P containing n members, where each member has only two sites, is there a haplotype configuration with at most k recombination events explaining P?*

There is a correspondence between an optimization version and a decision version of the MRHC problem. We can get a result for the optimization version of the problem by trying parameter k with 0 and increasing its value step by step to solve the decision version until the problem answer is Yes. On the other hand, we can immediately get a result for the decision version of the problem from a result of the optimization version.

3 Methods

We construct a pedigree graph to represent the 2-site-MRHC$_k$ problem.

3.1 Label Members

Given a member u and its two sites i and j, if sites i and j are both heterozygous or both homozygous, the member is *labeled*. If only one site is homozygous and the other site is heterozygous, the member is *unlabeled*.

If i and j are both homozygous with the same value ($g_u[i] = g_u[j] = 0$ or $g_u[i] = g_u[j] = 1$), u is labeled *green*. If i and j are both homozygous with different values ($g_u[i] = 0$ and $g_u[j] = 1$, or $g_u[i] = 1$ and $g_u[j] = 0$), u is labeled *red*. If i and j are both heterozygous, $g_u[i] = g_u[j] = 2$, u is labeled *grey*. A member is *resolved* if it is labeled red or green. A member is *unresolved* if it is labeled grey. A grey member u will later be resolved green if $h1_u[i] = h1_u[j] = 0$ or $h1_u[i] = h1_u[j] = 1$. It is resolved red otherwise. The resolution of a grey member depends on its adjacent members.

3.2 Insert Positive Edges

If u is a parent of v and both u and v are labeled, we insert a *positive edge*, $e_{pos}(u, v)$, between u and v. A positive edge $e_{pos}(u, v)$ means the label of u and the label of v should be the same once resolved, unless a recombination occurs in u. The reason for this is that if there is no recombination in u, then v receives one full haplotype from u and another full haplotype from another parent based on the Mendelian law of inheritance. Therefore, the label of u and the label of v should be the same if there is no recombination; otherwise, there is a recombination event in u.

3.3 Insert Negative Edges

We also consider a trio with two parents, u and v, and a child c. If both parents are labeled but the child is not labeled, we insert a *negative edge*, $e_{neg}(u, v)$, between u and v. A negative edge $e_{neg}(u, v)$ means u and v should be resolved with different labels, unless there is a recombination event in one parent of c.

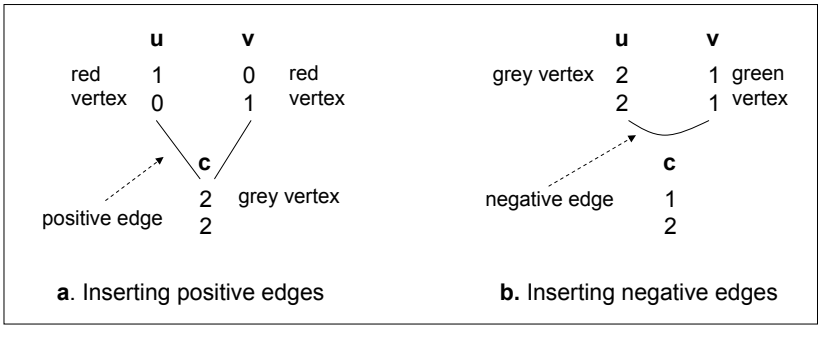


Fig. 2. Inserting positive and negative edges. Here genotypes of members are displayed.

This phenomenon can be explained as follows. If there is no recombination and u and v have the same resolved label, i.e., both red or both green, then sites i and j of c must be both homozygous or both heterozygous based on the Mendelian law of inheritance. Because only one site of c is homozygous and the other site is heterozygous, one recombination occurs if u and v have the same label when resolved, but no recombination occurs if they are resolved differently.

Figure 2 shows positive and negative edges inserted between members in the pedigree.

3.4 Process Unlabeled Members

So far, we have processed labeled members. Now we process an unlabeled member u that has one homozygous site and one heterozygous site.

If u is a child in its previous generation, a negative edge is inserted between two parents of u as discussed in Subsection 3.3. If u is a parent of a child c, there is no way to detect whether there is a recombination event in u caused by haplotype shuffling or not. This fact can be explained as follows. Without loss of generality, suppose $g_u[i] = 0$ and $g_u[j] = 2$, the haplotype pair of u inferred would be $h1_u = 01$ and $h2_u = 00$. The possible mixed haplotypes transferred to c from u are still either $\{01\}$ or $\{00\}$. In both cases, we can explain u as a member with no recombination event by pointing the haplotype of c that is received from u to the appropriate haplotype of u.

Because we use unlabeled child members to insert negative edges only and there is no way detect haplotype shuffling in unlabeled parental members, we only consider members that are labeled from now on. Once labeled members are resolved, we can resolve unlabeled members accordingly.

3.5 Pedigree Graph

Pedigree P can be considered to be an undirected graph $G = (V, E)$. Each vertex $v \in V$ is a member with three possible labels, red, green, and grey. Each edge $e(u, v) \in E$ is either a positive edge, $e \in E_{pos}$, or a negative edge; $e \in E_{neg}$, $(E = E_{pos} \cup E_{neg})$. Graph G, set up this way, is a signed graph [16]. Let $N(u)$ be the set of adjacent vertices of u. Let $w(e)$ be the weight of edge e. If e is a positive edge, $w(e) = +1$. If e is a negative edge, $w(e) = -1$.

Observation 1. *There are at most n vertices and $O(n)$ edges in the pedigree graph.*

There are n members in the pedigree. A vertex is created for each member, except for unlabeled members with one site homozygous and one site heterozygous. Thus there are at most n vertices in the pedigree graph.

Except for external parents, a member has two positive edges linking it to two parents. Therefore, the number of edges in the graph is linear in the number of child members. If a member is an unlabeled member, the two positive edges linking two parents and the child are replaced by a negative edge between the two parents. Thus the number of edges in the pedigree graph is $O(n)$.

The 2-site-MRHC$_k$ problem can now be solved by determining if we can label every grey vertex in G either red or green such that if we partition the set of vertices V into (V_{red}, V_{green}) and let E^* be the set of edges between V_{red} and V_{green} then

$$\sum_{e \in E^* \cap E_{pos}} w(e) + \sum_{e \in E_{neg} \setminus E^*} \mid w(e) \mid \leq k \tag{1}$$

Given a pedigree graph, any two adjacent members linked by a positive edge should be in the same partition, and any two adjacent members linked by a negative edge should be in different partitions. Any edge whose constraint is not satisfied represents a recombination event between the two adjacent members, or, in the case of a negative edge having endpoints in the same partition, between one parent and the child. Equation 1 thus counts the number of recombination events in the whole pedigree and ensures that it is at most k.

This problem can be reduced to the problem of finding the line index of a signed graph [16].

3.6 Signed Graph

A graph $G = (V, E)$ is a *signed graph* if it has both positive and negative edges $(E = E_{pos} \cup E_{neg})$ [16], where $w(e_{pos}) = 1$ and $w(e_{neg}) = -1$. Let (V_1, V_2) be a partition of V, and E^* be the set of edges between V_1 and V_2. The *line index* of the cut (V_1, V_2) is defined as:

$$l(V_1, V_2) = \sum_{e \in E^* \cap E_{pos}} w(e) + \sum_{e \in E_{neg} \setminus E^*} |w(e)| \tag{2}$$

The line index of graph G is defined as:

$$l(G) = \min_{V_1 \subseteq V} l(V_1, V_2) \tag{3}$$

The corresponding decision version of finding the line index of graph G is defined as follows.

LineIndex$_k$: *Given a signed graph G and a positive integer k, is there a line index of G at most k?*

Clearly, the 2-site-MRHC$_k$ problem can be reduced to the LineIndex$_k$ problem. We will show that the LineIndex$_k$ problem can be reduced to the Bipartization by Edge Removal problem, a classic NP-complete problem that is fixed-parameter tractable.

4 Fixed-Parameter Algorithm

A NP-hard problem cannot be solved by a polynomial time algorithm unless P=NP. However, if we can restrict some parameters of the problem to small

values, the running time of an algorithm for the problem can potentially be greatly reduced [3,11]. In this case, the problem is a *parameterized problem* and an algorithm that can solve the parameterized problem efficiently is a *fixed-parameter algorithm*. Formal definitions of parameterized problem and fixed-parameter algorithm [11] are as follows.

Definition 1. *A parameterized problem is a language $L \subseteq \Sigma^* \times \Sigma^*$, where Σ is a finite alphabet. The second component is called the parameter of the problem.*

Practically, the parameter is a nonnegative integer or a set of nonnegative integers and therefore $L \subseteq \Sigma^* \times \mathbb{N}$. For $(x, k) \in L$, the size of the input is $n = |(x, k)|$, and the parameter is k.

Definition 2. *A parameterized problem L is a fixed-parameter tractable if it can be determined in $f(k) \cdot n^{O(1)}$ time whether or not $(x, k) \in L$, where f is a computable function only depending on k. The corresponding class of problems is called FPT.*

A comprehensive survey of FPT problems can be found in [3] and [11].

4.1 Transforming to Bipartization by Edge Removal Problem

We review an important property of a signed graph given by [16].

Theorem 1. *Let G be a signed graph. If we replace each edge with weight $w(e) > 0$ by two consecutive edges with weight -w(e) to get a graph G' then $l(G) = l(G')$.*

Proof. Suppose (V_1, V_2) is a cut of G such that $l(V_1, V_2) = l(G)$. We replace each positive edge $e(u, v)$ by two consecutive negative edges $e(u, y)$ and $e(y, v)$, where $w(e(u, y)) = w(e(y, v)) = -w(e(u, v))$ and y is a new vertex adjacent only to u and v. If u and v belong to the same partition we put y in a different partition from the partition of u and v. If u and v belong to different partitions, we arbitrarily put y in the same partition of either the partition of u or v. In all of the cases above we find the corresponding cut of G', (V_1', V_2') such that $l(V_1', V_2') = l(V_1, V_2)$. Therefore $l(G') \leq l(G)$.

Conversely, if $l(V_1', V_2') = l(G')$ and y is a new vertex, then at least one edge incident to y is in the cut. We can find a corresponding cut of G, (V_1, V_2) such that $l(V_1, V_2)$=$l(V_1', V_2')$. Therefore $l(G') \geq l(G)$. Taken together, we get $l(G') = l(G)$.

Based on this property, the pedigree graph is transformed into a new graph by replacing every positive edge by two consecutive negative edges and adding new intermediate vertices. We obtain a new weighted graph G' with all negative weighted edges. The graph G' still has only $O(n)$ vertices and $O(n)$ edges. Equation 1 becomes

$$\sum_{e \in E_{neg} \setminus E^*} |w(e)| \leq k \tag{4}$$

This equation is to ensure that the total number of edges within V_1 and edges within V_2 is at most k. These edges once removed will make the graph bipartite.

We further transform our negative graph into a new graph with all positive edges by multiplying the weight of every edge by -1. Our problem becomes the Bipartization by Edge Removal problem [13,14]. The k-Bipartization by Edge Removal problem is defined as follows.

Definition 3. *Given a graph G=(V,E) and a positive integer k, is there a set* $C \subseteq E$ *with* $|C| \leq k$ *whose removal produces a bipartite graph?*

Bipartization by Edge Removal is a classical NP-hard problem and is in FPT [13,14]. Its parametric dual is Max-Cut [6].

4.2 FPT Algorithm for Bipartization by Edge Removal

One efficient technique to tackle an FPT problem is *iterative compression*. It is first proposed by [13] in a breakthrough paper and has been shown to very useful for solving different minimization problems. The idea is that, given a solution of size $(k+1)$, we find a fixed-parameter algorithm that either constructs a solution of size k if one exists or outputs No if no solution exists. We iteratively compress the problem by reducing the size of its solutions step by step. Assuming the running time of the FPT algorithm is $O(f(k) \cdot n^{O(1)})$, the overall running time will be $O(n \cdot f(k) \cdot n^{O(1)})$.

Iterative compression technique is used by Guo et al. [4] to solve the Bipartization by Edge Removal problem with a running time of $O(2^k \cdot m^2)$, where k is the number of edges to be deleted to make the graph bipartite.

Theorem 2. *The 2-site-MRHC$_k$ problem is solvable in* $O(2^k \cdot n^2)$ *time.*

Proof. Setting up the pedigree graph takes $O(|V|)$ time. Transforming the pedigree graph into a graph with all negative edges takes $O(|E|)$ time and transforming the negative graph into a graph with all positive edges takes $O(|E|)$ time. The Bipartization by Edge Removal problem can be solved in $O(2^k \cdot m^2)$. Our graph is sparse with the number of edges linear in the number of vertices, so the overall running time of our algorithm is $O(2^k \cdot n^2)$.

5 Conclusion

We have shown that the MRHC problem for general pedigrees with two sites can be reduced to the line index of a signed graph, and the line index of a signed graph can, in turn, be reduced to the Bipartization by Edge Removal problem. Therefore we can solve the MRHC problem for general pedigrees with two sites with an $O(2^k \cdot n^2)$ fixed-parameter algorithm. Future work will extend the current method to deal with genetic data with more than two sites.

References

1. Chan, B.M.-Y., Chan, J.W.-T., Chin, F.Y.-L., Fung, S.P.-Y., Kao, M.-Y.: Linear-time haplotype inference on pedigrees without recombinations. In: Bücher, P., Moret, B.M.E. (eds.) WABI 2006. LNCS, vol. 4175, pp. 56–67. Springer, Heidelberg (2006)
2. Doan, D.D., Evans, P.A., Horton, J.D.: A Near-Linear Time Algorithm for Haplotype Determination on General Pedigrees. Submitted to Journal of Computational Biology (2009) Submission ID: JCB-2009-0133
3. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Heidelberg (1999)
4. Guo, J., Gramm, J., Hüffner, F., Niedermeier, R., Wernicke, S.: Compression-Based Fixed-Parameter Algorithms for Feedback Vertex Set and Edge Bipartization. Journal of Computer and System Sciences 72(8), 1386–1396 (2006)
5. Gusfield, D.: An overview of combinatorial methods for haplotype inference. In: Istrail, S., Waterman, M.S., Clark, A. (eds.) DIMACS/RECOMB Satellite Workshop 2002. LNCS (LNBI), vol. 2983, pp. 9–25. Springer, Heidelberg (2004)
6. Karp, R.M.: Reducibility Among Combinatorial Problems. In: Miller, R.E., Thatcher, J.W. (eds.) Complexity of Computer Computations, pp. 85–103. Plenum, New York (1972)
7. Li, J., Jiang, T.: Efficient inference of haplotypes from genotypes on a pedigree. Jornal of Bioinformatics and Computational Biology 1(1), 41–70 (2003)
8. Li, J., Jiang, T.: Efficient rule-based haplotyping algorithms for pedigree data. In: Proceedings of Research in Computational Molecular Biology (RECOMB 2003), pp. 197–206. ACM Press, New York (2003)
9. Li, J., Jiang, T.: An exact solution for finding minimum recombinant haplotype configurations on pedigrees with missing data by integer linear programming. In: Proceedings of Research in Computational Molecular Biology (RECOMB 2004), pp. 20–29. ACM Press, New York (2004)
10. Liu, L., Chen, X., Xiao, J., Jiang, T.: Complexity and approximation of the minimum recombinant haplotype configuration problem. Theoretical Computer Science 378, 316–330 (2007)
11. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press, Oxford (2006)
12. Qian, D., Beckmann, L.: Minimum-recombinant haplotyping in pedigrees. American Journal of Human Genetics 70(6), 1434–1445 (2002)
13. Reed, B., Smith, K., Vetta, A.: Finding odd cycle transversals. Operations Research Letters 32, 299–301 (2004)
14. Wernicke, S.: On the algorithmic tractability of single nucleotide polymorphism (SNP) analysis and related problems. Diplomarbeit, Wilhelm-Schickard-Institut für Informatik, Universität Tübingen (2003)
15. Xiao, J., Liu, L., Xia, L., Jiang, T.: Fast elimination of redundant linear equations and reconstruction of recombination-free mendelian inheritance on a pedigree. In: Proceedings of the eighteenth annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), pp. 655–664 (2007)
16. Xu, S.: The line index and minimum cut of weighted graphs. European Journal of Operational Research 109, 672–682 (1998)
17. Zhang, K., Sun, F., Zhao, H.: HAPLORE: a program for haplotype reconstruction in general pedigrees without recombination. Bioinformatics 21(1), 90–103 (2005)

Analysis of Temporal-spatial Co-variation within Gene Expression Microarray Data in an Organogenesis Model

Martin Ehler[1,2], Vinodh Rajapakse[2], Barry Zeeberg[3], Brian Brooks[4], Jacob Brown[4], Wojciech Czaja[2], and Robert F. Bonner[1]

[1] National Institutes of Health, Eunice Kennedy Shriver National Institute of Child Health and Human Development, Section on Medical Biophysics, Bethesda MD 20892
[2] University of Maryland, Department of Mathematics, Norbert Wiener Center, College Park MD 20742
[3] National Institutes of Health, National Cancer Institute, Laboratory of Molecular Pharmacology, Genomics & Bioinformatics Group, Bethesda MD 20892
[4] National Institutes of Health, National Eye Institute, Ophthalmic Genetics and Visual Function Branch, Bethesda MD 20892

Abstract. The gene networks underlying closure of the optic fissure during vertebrate eye development are poorly understood. We used a novel clustering method based on Laplacian Eigenmaps, a nonlinear dimension reduction method, to analyze microarray data from laser capture microdissected (LCM) cells at the site and developmental stages (days 10.5 to 12.5) of optic fissure closure. Our new method provided greater biological specificity than classical clustering algorithms in terms of identifying more biological processes and functions related to eye development as defined by Gene Ontology at lower false discovery rates. This new methodology builds on the advantages of LCM to isolate pure phenotypic populations within complex tissues and allows improved ability to identify critical gene products expressed at lower copy number. The combination of LCM of embryonic organs, gene expression microarrays, and extracting spatial and temporal co-variations appear to be a powerful approach to understanding the gene regulatory networks that specify mammalian organogenesis.

Keywords: laser capture microdissection, microarray, organogenesis, gene regulatory network, clustering.

1 Introduction

Common variations in genetic and epigenetic patterns among humans are associated with variations in risk for developing all common chronic diseases, a few of which have been identified from genome-wide polymorphism screens [13,20]. The functional biological robustness or its failure in disease is most likely not just reflected in a few dominant components, but in many complex interactions within gene regulatory networks. Due to the overwhelming complexity, the

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 38–49, 2010.

deeper understanding of such networks remains a major challenge in modern systems biology that aims to discover and iteratively refine mechanistic models of biological processes. Biological knowledge is typically encoded in the structure and parameterization of these models. The Gene Ontology project [1,7] can help to incorporate the known biological details of gene functions into such analysis. The challenge is to reasonably approximate attributes in such models using experimental data that is complex, noisy, and often incomplete.

For the purpose of acquiring biologically rich data sets, laser capture microdissection (LCM) has proven a powerful tool to isolate pure cell populations from complex heterogeneous tissue specimens [4,8,18]. In combination with microarray technologies, that allow the simultaneous measurement of expression levels for thousands of genes, LCM enables identifying critical gene products even if expressed at low copy numbers.

Our work aims to facilitate efforts in systems biology by organizing data in ways that can potentially suppress noise and better reveal latent, biologically meaningful structure. Coloboma is a not uncommon congenital defect of human ocular development resulting in large retinal holes which often significantly affect vision. The present paper focuses on refinements in the analysis of a temporal series of microarray data obtained from microdissected sites of retinal fissure closure in normal mouse embryos. This data was previously analyzed [5] to identify a putative repressive transcription factor, nlz2 (zinc finger protein 503), which, when its expression was blocked in zebrafish embryos, led to incomplete optic fissure closure, a coloboma model. By developing and applying a novel clustering scheme, we have identified a 50 per cent larger gene cluster (in comparison to PCA and previous hierarchical cluster analyses [5]), whose spatio-temporal gene expressions correlate with nlz2. According to GoMiner, a computational high-throughput tool for biological interpretation of genomic, transcriptomic, and proteomic data, that identifies the biological processes, functions and components of gene clusters [21,22], this larger cluster still shows gene enrichment for its specific functions in the context of Gene Ontology.

Next, using GoMiner, we sought to identify those gene clusters whose co-expressions correlate with processes in eye development. We apply a novel clustering scheme that builds on the intertwining of Laplacian Eigenmaps, a geometrical data transformation, with k-means, a standard clustering method. To validate the findings, we also use two standard clustering schemes, basic k-means and principal component analysis combined with k-means. All three methods identify gene clusters enriched for functional GoMiner categories related to eye development, but our proposed novel scheme leads to lower false discovery rates. In this sense, our new clustering method appears to provide greater biological specificity and sensitivity.

Starting from experimental work based on LCM and microarray technologies in organogenesis, we obtained a list of candidate genes that could be significant in normal development of optic fissure closure and could be useful in guiding analysis of genetic variations in humans with coloboma.

2 Materials and Methods

The Affymetrix MOE 430 2.0 microarray datasets analyzed to develop and test our new method were for eight samples LCM microdissected from serial cryosections of the retina at the site of final optic fissure closure in the mouse embryos at specific embryonic stages 10.5D through 12.5D previously reported in [5]. The 8-timepoints span the time just before and just after final fusion (optic fissure closure) and were expected to reveal sets of genes critical for the completion of optic fissure closure in normal development. This previous report further investigated a specific putative repressive transcription factor, nlz2 or zinc finger protein 503, that was discovered to be highly expressed before during fissure closure and then downregulated. Gene knockdown experiments in zebra fish of nlz2 resulted in incomplete optic fissure closure (coloboma). Our current analysis explored possible associated gene regulation patterns. Within the 8 different time-point microarrays were 8316 genes consistently identified as expressed and with greater than 2-fold variation in gene expression levels. For our clustering analysis, we chose the subset of 3416 genes whose expression levels varied between 4-fold and 26-fold over the 2 days of embryonic development.

For analysis purposes, each gene of the microarray is considered as a vector of its expression levels. This perspective yields a collection of $D = 8$ dimensional vectors. Our proposed analysis relies on Laplacian Eigenmaps, cf. Section 2.2, a geometrical data transformation that provides a new representation of gene expressions still covering essential geometrical behaviors. We intertwine this new data representation with k-means, cf. Section 2.3, a widely used clustering scheme. GoMiner, cf. Section 2.4, is then used to identify genes within clusters that are associated with particular biological processes or function. Let us list the steps of our proposed scheme:

1. **Expression Vectors:** Each gene's expression over the 8 time points builds a vector. They constitute a collection $\{x_1, \ldots, x_n\}$ of 8-dimensional vectors.
2. **Laplacian Eigenmaps:** Choose the number m of gene neighbors and a target dimension d, then apply Laplacian Eigenmaps to obtain a new data representation $\{y_1, \ldots, y_n\}$ of d-dimensional vectors.
3. **k-means:** Run k-means on $\{y_1, \ldots, y_n\}$ to obtain the final clustering.
4. **GoMiner:** Feed the clusters into GoMiner to evaluate their biological relevance.

In the following, we present the components of the above scheme in more detail. For comparison we also applied PCA and k-means and therefore briefly discuss these conventional methods too.

2.1 Principal Component Analysis

PCA [14] is a statistical tool that linearly transforms the data into an orthogonal coordinate system whose axes correspond to the principal components in the data, i.e., the first principal component accounts for as much variance in the data as possible and, successively, further components capture the remaining

variance. Through an eigenanalysis, the principal components are determined as eigenvectors of the dataset's covariance matrix and the corresponding eigenvalues refer to the variance that is captured within each eigenvector. After subtracting the mean of the dataset, PCA is performed on vectors $\{x_1, \ldots, x_n\}$ by first diagonalizing the covariance matrix $\mathrm{cov}(X) = E(XX^\top)$, where $X = (x_1 \cdots x_n)$ is the zero mean data matrix. The eigenvectors $p_1, \ldots, p_D$ - the principal components ordered according to the magnitude of their eigenvalues - provide the transformed data $Y = W^\top X$, where $W = (p_1 \ldots p_D)$. We obtain the collection of d-dimensional vectors $\{y_1, \ldots, y_n\}$ whose first entries represents the abundance of the primary principal. The second entries are each datapoint's projection along the second eigenvector and so forth.

2.2 Laplacian Eigenmaps

Laplacian Eigenmaps (LE) [2,3] is a geometric tool that transforms data into a new representation in a nonlinear fashion. Given points $\{x_1, \ldots, x_n\} \subset \mathbb{R}^D$, we assume that they are steered by d latent variables, and aim to find a new data representation $\{y_1, \ldots, y_n\} \subset \mathbb{R}^d$. We briefly recall the three step procedure of Laplacian Eigenmaps.

Step 1: Adjacency graph, *m*-nearest neighbors. We build a graph $\mathcal{G}$, whose nodes i and j are connected if x_i is among the m-nearest neighbors of x_j or vice versa. The distance between data points is measured by the Euclidean metric. The graph $\mathcal{G}$ represents the connectivity of the data vectors.

Step 2: Heat kernel as weights. Next, we weight the edges of the graph and focus on the diffusion *weight matrix* W given by

$$W_{i,j} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{\sigma}}, & i \text{ and } j \text{ are connected,} \\ 0, & \text{otherwise.} \end{cases}$$

The number of neighbors m controls the sparsity of W.

Step 3: Solving an eigenvalue problem. We denote a potential new data representation by $y = (y_1, \ldots, y_n)^\top$, where each row is considered as a vector in $\mathbb{R}^d$, and we then consider the following minimization problem

$$\min_{y^\top Dy = E} \tfrac{1}{2} \sum_{i,j} \|y_i - y_j\|^2 W_{i,j} = \min_{y^\top Dy = E} \mathrm{trace}(y^\top L y), \tag{1}$$

where $L = D - W$ and D is the diagonal matrix $D_{i,i} = \sum_j W_{i,j}$. The minimizer of (1) is given by the d minimal eigenvalue solutions of $Lx = \lambda Dx$ under the constraint $y^\top Dy = E$, i.e., the minimizer y's columns are the d eigenvectors with respect to the smallest eigenvalues. If the graph is connected, then $\mathbf{1} = (1, \ldots, 1)^\top$ is the only eigenvector with eigenvalue 0, and we exclude it. Instead of (1), we try to find the minimizer of

$$\min_{\substack{y^\top Dy = E, \\ y^\top D\mathbf{1} = 0}} \mathrm{trace}(y^\top L y). \tag{2}$$

By applying the change of variables $z = D^{1/2}y$, this yields

$$\min_{\substack{z^\top z = E, \\ z^\top \mathbf{1} = 0}} \operatorname{trace}(z^\top \mathcal{L} z), \tag{3}$$

where $\mathcal{L} = D^{-1/2} L D^{-1/2}$. The minimizer z is given by the d eigenvectors with smallest nonzero eigenvalue, and we obtain the d-dimensional representation $\{y_1, \ldots, y_n\}$ from $y = D^{-1/2}z$.

2.3 Standard Cluster Analysis

For hierarchical clustering, we refer to [9], and we have also applied a shape similarity-based clustering as introduced in [10]. k-means is a method of cluster analysis which aims to partition n observations into k clusters $\{c_1, \ldots, c_k\}$, where k has to be chosen a-priori [12], i.e., one aims at minimizing

$$\arg \min_{c_1,\ldots,c_k} \Big(\sum_{j=1}^{k} \sum_{y_i \in c_j} \|y_i - Ec_j\|^2 \Big),$$

where Ec_j is the mean of cluster c_j. The basic k-means algorithm requires the target number of clusters to be specified as a parameter.

The k-means algorithm begins with a data set, a target number of clusters k, and a set of $s_1, \ldots, s_k$ initial cluster centroids. It then iteratively assigns points to clusters by centroid proximity, and then adjusts centroids to reflect changes in cluster membership. The algorithm terminates either after a specified number of iterations, or once the cluster centroids/membership no longer change. Although optimal results cannot be guaranteed, the algorithm is quite fast, and many runs can be efficiently computed, with the best clustering taken as an overall result.

2.4 GoMiner

GoMiner provides a quantitative and statistical analysis-tool for biological interpretation of genomic, transcriptomic, and proteomic data, commonly derived from gene expression microarray experiments. It classifies genes into biologically coherent categories and then uses the Gene Ontology project to identify the biological processes, functions and components of genes within these categories [21,22]. A one-sided Fisher's p-value is used to determine the significance and biological enrichment levels within a category.

2.5 Clustering with Genesis

Clustered image maps (CIMs) were first introduced in [19] and were produced here with the Genesis program [17]. We selected the Euclidean distance metric and average linkage for hierarchal clustering. To facilitate visualization, we implemented a recently-added feature of GoMiner that removes large generic categories from all CIMs.

2.6 Silhouette Coefficient

The silhouette coefficient is a measure for the coherence of clusters. If we take a clustering C to be a mapping from a data set $X = \{x_1, \ldots, x_n\}$ to the integers $1, 2, \ldots, k$ (where k is the total number of clusters), we can define the silhouette coefficient $sil(x)$ for each point x in X to be

$$sil(x) = \frac{B(x) - A(x)}{\max(A(x), B(x))},$$

where $A(x)$ is the average distance between x and other points in its cluster, and $B(x)$ is the minimum distance between x and the nearest neighboring cluster, cf. [11]. The silhouette coefficient $sil(i)$ for a cluster i is the average of the coefficients for its constituent points. We similarly define the silhouette coefficient sil for an entire clustering to the average silhouette coefficient over all data set points. A clustering with a silhouette coefficient closer to 1 will contain more cohesive and well-separated clusters.

For our experiments, we used the squared Euclidean distance for the computations indicated above, as well as for the data clustering algorithms.

Supplementary material is available under:
http://discover.nci.nih.gov/RetDev/supplementaryMaterials.html [6].

3 Results

We aim to increase our understanding of the gene network underlying the closure of the optic fissure during vertebrate eye development. Microarray data from LCM isolated cells in a mouse model of coloboma as described in Section 2 were analyzed by using standard cluster analysis and a novel gene clustering scheme. We derive a coherent clustering and make use of GoMiner to identify those genes identified in public databases as being associated with eye development or function as a measure of the quality of the other members in the cluster. The newly identified genes are then potentially higher quality candidates for association with retinal development at these stages and, in particular, closure of the optic fissure.

For k-means, we set the target number of clusters to be 24, based on previous work with the current data set [5] that yielded biologically meaningful (but smaller and fewer) cluster results. The maximal silhouette coefficient sil specifies the best k-means clustering over 100 repeated runs, starting in each case from different randomly selected initial centroids. The maximum was stable over different 100 run sets, suggesting that an at least near optimal clustering was being obtained. Since the parameter space is too big for an exhaustive search in Laplacian Eigenmaps, we fixed $\sigma = 1/8$ and assessed remaining parameters over $m = 5, \ldots, 10, 12, 15, 20, 25, 50, 100$ and $d = 1, \ldots, 10, 12, 16$. The idea is that parameter combinations that yield better cluster structure in the mapped

data $\{y_1, \ldots, y_n\}$ might be better tuned to resolve possible intrinsic structure in the original data $\{x_1 \ldots, x_n\}$. Silhouette coefficients suggest $m = 10$ and $d = 2$ additionally providing excellent GoMiner gene identifications.

Enlarged cluster containing nlz2: We have identified a 50 per cent larger gene cluster than with hierarchical clustering in [5] whose spatio-temporal gene expressions significantly correlate with nlz2, a gene which when previously inhibited in zebrafish induced coloboma. The latter cluster was associated with 210 Affymetrix probes corresponding to 169 genes, nlz2 was among them. See Figure 1 for gene expression profiles and its set of enriched functional categories. GoMiner assigned the functional category of 'gene silencing', indicating the repressive influence of nlz2 and co-varying genes. Previous biological studies have shown nlz2 gene product to repress gene transcription of a number of genes regulated hindbrain development possibly as part of a transcription factor complex consistent with its H2N2 zinc finger domain and its binding site for histone deacetylase. Consistent with this hypothesis, we also identified an additional cluster that varied inversely with the primary 'nlz2 cluster' gene silencing, suggestive of the previously documented role of nlz2 in suppression of gene transcription, cf. Figure 3.

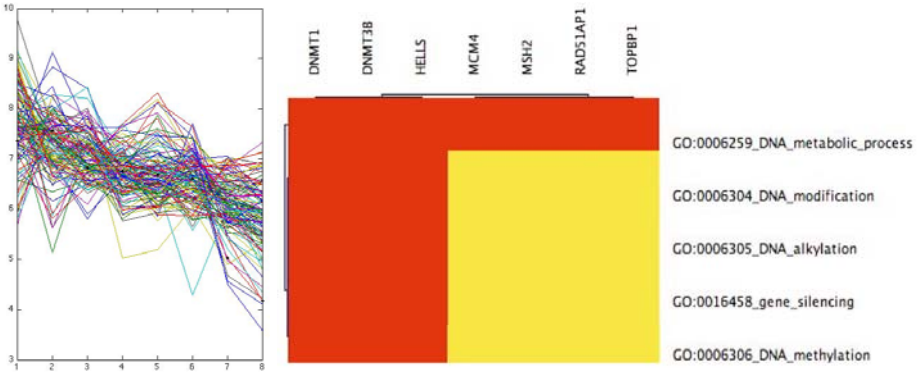


Fig. 1. Cluster containing nlz2: (left) cluster profile, i.e., gene expression levels vs. 8 time points, black circles indicate nlz2, (right) enriched functional categories

One complementary cluster: We found a large cluster whose shape is distinct from nlz2 by applying the similarity-based shape clustering in [10]. GoMiner assigned a number of significantly associated functions to this large cluster including **retina morphogenesis** (vertebrate eye), **generation of neurons**, cellular morphogenesis during differentiation, photoreceptor differentiation, cell motility, **neuron differentiation**, cell projection organization, and biogenesis. The highlighted functions are specifically associated with CHX10, a gene in this cluster that has previously been identified in retinal development, see, for instance, [15,16].

Collection of enriched clusters: We also applied k-means on the original data set and on PCA and LE reduced data. The selected 'best' k-means result applied directly to the original data had an overall silhouette coefficient of 0.38. To evaluate PCA+k-means, for each possible number of retained principal components, the mapped data were clustered, and overall silhouette scores were obtained. The best results refer to the mapping based on just the first principal component, with the best overall silhouette score being 0.698. The silhouette scores in the mapped data were always substantially higher than those obtained following clustering of the original data, in keeping with the idea that Laplacian Eigenmaps can potentially enhance cluster structure, see Table 1 for more details. We found that PCA+k-means, basic k-means, and LE+k-means yielded

Table 1. Silhouette coefficients and number of genes for each cluster/clustering method

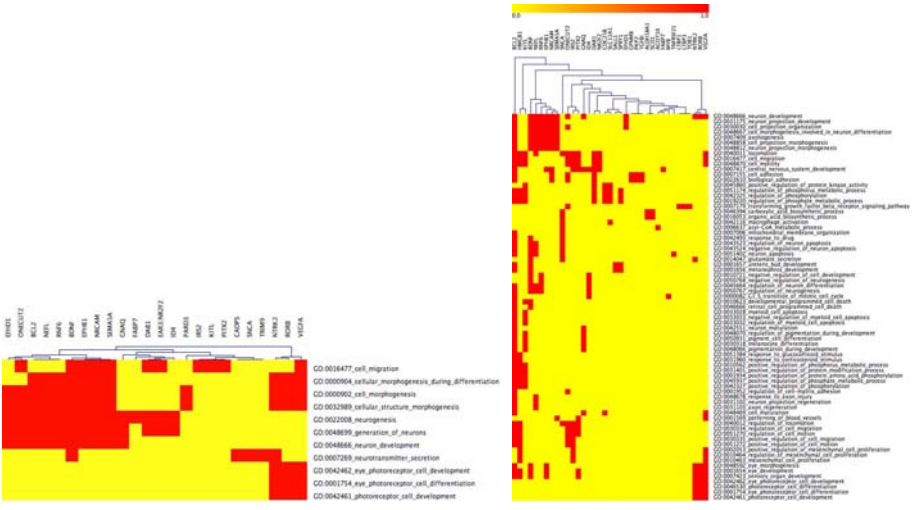
	k-means		PCA+k-means		LE+k-means	
cluster	sil	# genes	sil	# genes	sil	# genes
1	0.0200	65	0.7329	126	0.6535	103
2	0.3067	146	0.6221	60	0.7049	125
3	0.4078	180	0.7002	168	0.6862	174
4	0.4068	234	0.6840	198	0.6848	154
5	0.3401	255	0.7423	157	0.7831	97
6	0.2960	252	0.7033	130	0.7949	389
7	0.3442	90	0.6795	126	0.7369	120
8	0.6509	9	0.6800	65	0.6953	270
9	0.3900	254	0.6393	190	0.7800	91
10	0.2162	34	0.7130	187	0.7046	79
11	0.3056	112	0.6517	182	0.7606	141
12	0.3531	165	0.7162	155	0.7487	122
13	0.4636	182	0.6925	117	0.9889	3
14	0.4267	167	0.7422	205	0.7118	125
15	0.6529	114	0.6968	184	0.5997	85
16	0.1593	86	0.5266	9	0.7214	236
17	0.5488	13	0.6792	84	0.6839	83
18	0.4323	253	0.6956	211	0.7380	135
19	0.1749	20	0.7151	118	0.6466	72
20	0.3076	133	0.6926	170	0.7243	121
21	0.4314	174	0.7041	115	0.7461	199
22	0.4394	130	0.7342	116	0.7442	275
23	0.4538	210	0.7252	192	0.6849	115
24	0.4366	138	0.6792	151	0.8534	102

several significantly enriched gene clusters (out of a total of 24) associated with developmental processes. Cluster 22 of the Laplacian Eigenmaps-based approach revealed a cluster significantly enriched (with a false discovery rate (FDR) of less than 0.05) for genes specifically implicated in eye development - which was the focus of the experimental work underlying the data set considered in this study. These functional categories (in GoMiner terminology) were

(i) GO:0042462_eye_photoreceptor_cell_development,
(ii) GO:0001754_eye_photoreceptor_cell_differentiation,
(iii) GO:0042461_photoreceptor_cell_development.

When slightly relaxing the FDR up to < 0.15, this cluster 22 shows gene enrichment for further eye specific developmental functions:

(iv) GO:0048592_eye_morphogenesis,
(v) GO:0001654_eye_development,
(vi) GO:0046530_photoreceptor_cell_differentiation,



(a) FDR< 0.05 (b) FDR< 0.15, increase in functional associations

Fig. 2. CIM thumbnails for LE+k-means cluster 22 related to eye development, go to `http://discover.nci.nih.gov/RetDev/supplementaryMaterials.html` [6] for full-size CIMs

see also Figure 2. These categories are neither hit by k-means nor PCA+k-means clustering when restricting the FDR to < 0.05. By relaxing the FDR, however, both k-means and PCA+k-means clustering show gene enrichment for eye specific functions. This verifies that the eye specific functions in LE+k-means cluster 22 are real and have not been picked up by chance. To support the latter claim, we have compared the enriched categories in the LE+k-means cluster 22 with the clusters of the other two clustering methods with relaxed FDR. It turns out that specific eye development functions are present in all three clustering methods, but our proposed Laplacian-based scheme leads to lower false discovery rates and hence appears to provide greater biological specificity and sensitivity, see the supplementary material (`http://discover.nci.nih.gov/RetDev/supplementaryMaterials.html`) [6]. CIMs in these supplements indicate which clusters across the three methods share common GoMiner categories. It enables us to identify categories that are more specific to one method than to the others. Based on Table 1 the fraction of genes, that are associated to biological functions, are computable for each cluster, method, and false discovery rate.

Note on LE+k-means: We noted that relatively unusual expression patterns were often mapped to distinct, outlying clusters by the Laplacian Eigenmaps approach. For example, the three expression patterns indicated in Figure 3 formed a distinct cluster under the Laplacian Eigenmaps data representation. They

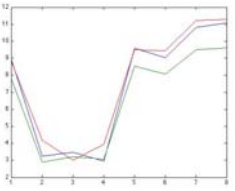


Fig. 3. Outliers that LE+k-means captures into a separate cluster, the associated Affymetrix probes are 1427262_at, 1427263_at, 1436936_s_at

were not as well separated in the original and PCA-mapped data, and were consequently misplaced in inappropriate clusters. This could be a technical explanation for more biological specificity of the proposed clustering scheme based on Laplacian Eigenmaps.

4 Discussion

Obtaining a clearer understanding of the gene regulatory network underlying optic fissure closure during eye development will be a long process involving genetic analysis of humans with coloboma and studies of eye development in animal models. Our present analysis and results were focused on expanding a list of candidate genes that could be critical for normal fissure closure and in coloboma patients may contain mutations. Compared with conventional clustering algorithms that we tested, our new method was able to identify larger clusters associated either with the nlz2 gene expression or with a distinctly complementary pattern enriched with associations to eye development gene ontologies. It also uniquely identified the 'nlz2-repressed' pattern as a distinct cluster, cf. Figure 3. The large temporally covarying gene cluster in Figure 1 was identified by GoMiner as being significantly associated with gene silencing, suggestive of a gene regulatory network that represses alternative fates until optic fissure closure is successfully completed (day 11.5 in the mouse). The pattern of genes in Figure 3 could represent such genes that are transiently repressed only when the nlz2 cluster is high. Using temporal pattern-based similarity clustering [10] allowed identification of other distinct clusters (i.e., not containing nlz2) with other GoMiner identified significant associations with specific developmental functions in databases.

Clearly, our new mathematical approach to identify new components of gene regulatory networks controlling development is preliminary and biologically untested. Including additional databases of the associations among transcription factors and the genes whose expression they modulate would be valuable. Applying LCM, gene expression microarrays, and improvements in our analysis methods to mammalian organogenesis could be part of an iterative process to more completely identify additional elements in gene regulatory networks.

5 Conclusion

Microarray data are commonly used for global searches for gene expression changes that might be associated with a perturbation of a cell state or in pathology. In organ development, temporal and spatial patterns accessible through microdissection are associated with reproducible changes in gene expression of even larger numbers of genes. More efficient analysis of microarray data from such microdissected samples could provide improved understanding of cell fate and organogenesis as well as elaboration of gene expression covariance networks. Our analysis scheme based on Laplacian Eigenmaps appear to offer advantages over standard clustering algorithms in the sense of greater biological specificity and sensitivity.

Acknowledgments

The research was funded by the Intramural Research Program of NICHD/NIH, by NSF (CBET0854233), by NGA (HM15820810009), and by ONR (N000140910144).

References

1. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat. Genet. 25(1), 25–29 (2000)
2. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. NIPS 14 (2002)
3. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural. Comput. 15(6), 1373–1396 (2003)
4. Bonner, R.F., Emmert-Buck, M., Cole, K., Pohida, T., Chuaqui, R., Goldstein, S., Liotta, L.A.: Laser capture microdissection: molecular analysis of tissue. Science 21(278), 1481–1483 (1997)
5. Brown, J.D., Dutta, S., Bharti, K., Bonner, R.F., Munson, P.J., Dawid, I.B., Akhtar, A.L., Onojafe, I.F., Alur, R.P., Gross, J.M., Hejtmancik, J.F., Jiao, X., Chan, W., Brooks, B.P.: Expression profiling during ocular development identifies 2 nlz genes with a critical role in optic fissure closure. Proc. Nat. Acad. Sci. USA 106(5), 1462–1467 (2009)
6. Ehler, M., Rajapakse, V., Zeeberg, B., Brooks, B., Brown, J., Czaja, W., Bonner, R.F.: Supplementary materials: Analysis of temporal-spatial co-variation within gene expression microarray data in an organogenesis model (2010), `http://discover.nci.nih.gov/RetDev/supplementaryMaterials.html`
7. Gene Ontology Consortium: The gene ontology project in 2008. Nucleic Acids Res. 36(Database issue), D440–D444 (Janaury 2008)
8. Goldstein, S.R., McQueen, O.G., Bonner, R.F.: Thermal modeling of laser capture microdissection. Appl. Opt. 37(31), 7378–7391 (1998)
9. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer, Heidelberg (2009)

10. Hestilow, T., Huang, Y.: Clustering of gene expression data based on shape similarity. EURASIP J. Bioinform. Syst. Biol. (2009)
11. Kaufman, L., Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, London (1990)
12. Lloyd, S.P.: Least-squares quantization in PCM. IEEE Transactions On Information Theory 28(2), 129–137 (1982)
13. Mordantameron, D.J., Yang, Z., Gibbs, D., Chen, H., Kaminoh, Y., Jorgensen, A., Zeng, J., Luo, L., Brinton, E., Brinton, G., Bernstein, P.S., Brand, J.M., Zabriskie, N.A., Tang, S., Constantine, R., Tong, Z., Zhang, K.: Htra1 variant confers similar risks to geographic atrophy and neovascular age-related macular degeneration. Cell Cycle 6(9), 1122–1125 (2007)
14. Pearson, K.: On lines and planes of closest fit to systems of points in space. Philosophical Magazine 2(7-12), 559–572 (1901)
15. Reichman, S., Kalathur, R.K.R., Lambard, S., Aït-Ali, N., Yang, Y., Lardenois, A., Ripp, R., Poch, O., Zack, D.J., Sahel, J., Léveillard, T.: The homeobox gene CHX10/VSX2 regulates RdCVF promoter activity in the inner retina. Hum. Mol. Genet. 19(2), 250–261 (2010)
16. Sigulinsky, C.L., Green, E.S., Clark, A.M., Levine, E.M.: Vsx2/Chx10 ensures the correct timing and magnitude of hedgehog signaling in the mouse retina. Dev. Biol. 317(2), 560–575 (2008)
17. Sturn, A., Quackenbush, J., Trajanoski, Z.: Genesis: cluster analysis of microarray data. Bioinformatics 18(1), 207–208 (2002)
18. Suárez-Quian, C.A., Goldstein, S.R., Bonner, R.F.: Laser capture microdissection: a new tool for the study of spermatogenesis. J. Androl. 21(5), 601–608 (2000)
19. Weinstein, J.N., Myers, T.G., O'Connor, P.M., Friend, S.H., Fornace, J.A.J., Kohn, K.W., Fojo, T., Bates, S.E., Rubinstein, L.V., Anderson, N.L., Buolamwini, J.K., van Osdol, W.W., Monks, A.P., Scudiero, D.A., Sausville, E.A., Zaharevitz, D.W., Bunow, B., Viswanadhan, V.N., Johnson, G.S., Wittes, R.E., Paull, K.D.: An information-intensive approach to the molecular pharmacology of cancer. Science 275(5298), 343–349 (1997)
20. Yang, Z., Camp, N.J., Sun, H., Tong, Z., Gibbs, D., Cameron, D.J., Chen, H., Zhao, Y., Pearson, E., Li, X., Chien, J., Dewan, A., Harmon, J., Bernstein, P.S., Shridhar, V., Zabriskie, N.A., Hoh, J., Howes, K., Zhang, K.: A variant of the HTRA1gene increases susceptibility to age-related macular degeneration. Science 314(5801), 992–993 (2006)
21. Zeeberg, B., Feng, W., Wang, G., Wang, M.D., Fojo, A.T., Sunshine, M., Narasimhan, S., Kane, D.W., Reinhold, W.C., Lababidi, S., Bussey, K.J., Riss, J., Barrett, J.C., Weinstein, J.N.: GoMiner: a resource for biological interpretation of genomic and proteomic data. Genome Biol. 4(4), R28 (2003)
22. Zeeberg, B., Qin, H., Narasimhan, S., Sunshine, M., Cao, H., Kane, D.W., Reimers, M., Stephens, R.M., Bryant, D., Burt, S.K., Elnekave, E., Hari, D.M., Wynn, T.A., Cunningham-Rundles, C., Stewart, D.M., Nelson, D., Weinstein, J.N.: High-throughput GoMiner, an 'industrial-strength' integrative gene ontology tool for interpretation of multiple-microarray experiments, with application to studies of common variable immune deficiency (cvid). BMC Bioinformatics 6, 168 (2005)

Human Genome Annotation
(Invited Keynote Talk)

Mark Gerstein

Yale University
New Haven, CT 06520 USA

A central problem for 21st century science is annotating the human genome and making this annotation useful for the interpretation of personal genomes. My talk will focus on annotating the 99% of the genome that does not code for canonical genes, concentrating on intergenic features such as structural variants (SVs), pseudogenes (protein fossils), binding sites, and novel transcribed RNAs (ncRNAs). In particular, I will describe how we identify regulatory sites and variable blocks (SVs) based on processing next-generation sequencing experiments. I will further explain how we cluster together groups of sites to create larger annotations. Next, I will discuss a comprehensive pseudogene identification pipeline, which has enabled us to identify >10K pseudogenes in the genome and analyze their distribution with respect to age, protein family, and chromosomal location. Throughout, I will try to introduce some of the computational algorithms and approaches that are required for genome annotation. Much of this work has been carried out in the framework of the ENCODE, modENCODE, and 1000 genomes projects.

References

1. http://pseudogene.org
2. http://GenomeTECH.Gersteinlab.org
3. Balasubramanian, S., Zheng, D., Liu, Y.J., Fang, G., Frankish, A., Carriero, N., Robilotto, R., Cayting, P., Gerstein, M.: Comparative analysis of processed ribosomal protein pseudogenes in four mammalian genomes. Genome Biol. 10, R2 (2009)
4. Du, J., Bjornson, R.D., Zhang, Z.D., Kong, Y., Snyder, M., Gerstein, M.B.: Integrating sequencing technologies in personal genomics: optimal low cost reconstruction of structural variants. PLoS Comput. Biol. 5, e1000432 (2009)
5. Kim, P.M., Lam, H.Y., Urban, A.E., Korbel, J.O., Affourtit, J., Grubert, F., Chen, X., Weissman, S., Snyder, M., Gerstein, M.B.: Analysis of copy number variants and segmental duplications in the human genome: Evidence for a change in the process of formation in recent evolutionary history. Genome Res. 18, 1865–1874 (2008)
6. Korbel, J.O., Abyzov, A., Mu, X.J., Carriero, N., Cayting, P., Zhang, Z., Snyder, M., Gerstein, M.B.: PEMer: a computational framework with simulation-based error models for inferring genomic structural variants from massive paired-end sequencing data. Genome Biol. 10, 23 (2009)

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 50–51, 2010.

7. Lam, H.Y., Khurana, E., Fang, G., Cayting, P., Carriero, N., Cheung, K.H., Gerstein, M.B.: Pseudofam: the pseudogene families database. Nucleic Acids Res. 37, D738–D743 (2009)
8. Lam, H.Y., Mu, X.J., Stütz, A.M., Tanzer, A., Cayting, P.D., Snyder, M., Kim, P.M., Korbel, J.O., Gerstein, M.B.: Nucleotide-resolution analysis of structural variants using BreakSeq and a breakpoint library. Nat. Biotechnol. 28, 47–55 (2010)
9. Rozowsky, J., Euskirchen, G., Auerbach, R.K., Zhang, Z.D., Gibson, T., Bjornson, R., Carriero, N., Snyder, M., Gerstein, M.B.: PeakSeq enables systematic scoring of ChIP-seq experiments relative to controls. Nat. Biotechnol. 27, 66–75 (2009)
10. Snyder, M., Weissman, S., Gerstein, M.: Personal phenotypes to go with personal genomes. Mol. Syst. Biol. 5, 273 (2009)
11. Wang, L.Y., Abyzov, A., Korbel, J.O., Snyder, M., Gerstein, M.: MSB: A mean-shift-based approach for the analysis of structural variation in the genome. Genome Res. 19, 106–117 (2009)
12. Zhang, Z.D., Paccanaro, A., Fu, Y., Weissman, S., Weng, Z., Chang, J., Snyder, M., Gerstein, M.B.: Statistical analysis of the genomic distribution and correlation of regulatory elements in the ENCODE regions. Genome Res. 17, 787–797 (2007)
13. Zheng, D., Frankish, A., Baertsch, R., Kapranov, P., Reymond, A., Choo, S.W., Lu, Y., Denoeud, F., Antonarakis, S.E., Snyder, M., Ruan, Y., Wei, C.L., Gingeras, T.R., Guigo, R., Harrow, J., Gerstein, M.B.: Pseudogenes in the ENCODE regions: consensus annotation, analysis of transcription, and evolution. Genome Res. 17, 839–851 (2007)

Extensions and Improvements to the Chordal Graph Approach to the Multi-state Perfect Phylogeny Problem

Rob Gysel and Dan Gusfield

Department of Computer Science, University of California, Davis, 1 Shields Avenue, Davis CA 95616, USA
rsgysel@ucdavis.edu, gusfield@cs.ucdavis.edu

Abstract. The multi-state perfect phylogeny problem is a classic problem in computational biology. When no perfect phylogeny exists, it is of interest to find a set of characters to remove in order to obtain a perfect phylogeny in the remaining data. This is known as the character removal problem. We show how to use chordal graphs and triangulations to solve the character removal problem for an arbitrary number of states, which was previously unsolved. We outline a preprocessing technique that speeds up the computation of the minimal separators of a graph. Minimal separators are used in our solution to the missing data character removal problem and to the solution of the perfect phylogeny problem with missing data discussed in [10].

1 Introduction

An instance of the *k-state perfect phylogeny problem* (PP) is given by a matrix $M \in \{1, 2, \ldots, k\}^{n \times m}$, where each row corresponds to a *taxon* , each column a *character* , and the entry m_{ij} is the *state* that taxon i takes on character j. We wish to decide if a perfect phylogeny exists. A *perfect phylogeny* for M is a tree T where every node v is labeled by a vector $l(v) \in \{1, 2, \ldots, k\}^m$ and for each character j and state r, the subgraph of T induced by nodes v where $l_j(v) = r$ is connected. See also [9,10]. When M has missing entries, the *perfect phylogeny problem with missing data* (MD) asks whether we can impute values so that the resulting data has a perfect phylogeny. In many cases, it is impossible to construct a perfect phylogeny, and in such cases, we are interested in removing the minimum number of characters to find a new matrix that has a PP solution. This is called the *character removal problem* (CR). When M has missing data, finding the minimum number of characters to remove so that the remaining data has a MD solution is the *missing data character removal problem* (MDCR).

It is well known [16,8] that for binary data ($k = 2$), the CR problem reduces to the node-cover problem. An approach to MDCR for binary data is found in [11] and approaches to MDCR for $k = 3$ and CR for $k \leq 5$ states is found in [10]. Chordal graph theory was used in [10] to construct an algorithm to solve MD using minimal separators. In this paper we outline a preprocessing tool used to

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 52–60, 2010.

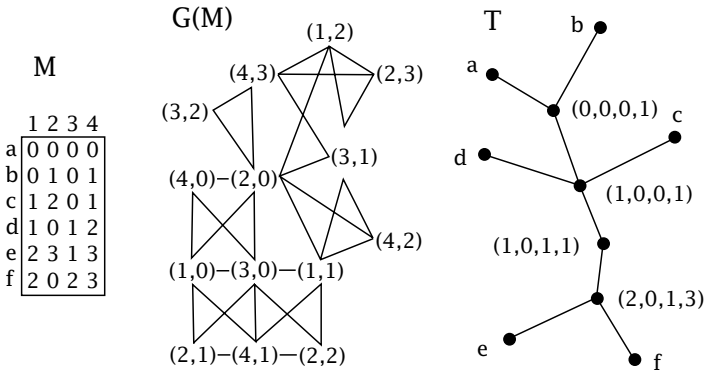


Fig. 1. An instance M of PP with perfect phylogeny T and PIG $G(M)$

calculate minimal separators faster, and show how to use chordal graph theory to solve MDCR. Section 3 provides a solution to MDCR using chordal graph theory. In Sect. 4 we outline the necessary theory for our preprocessing technique and Sect. 5 shows the empirical results in our preprocessing computations.

2 Preliminaries

We use $G = (V, E)$ to denote a graph with vertex set V and edge set E, along with n and m to refer to the number of vertices and edges of G, respectively. When necessary, we use $V(G)$ and $E(G)$ to denote the vertex and edge set of G, respectively. Two vertices u and v are *adjacent* if uv is an edge. A collection of vertices U is a *clique* if every pair of vertices in U are adjacent. The *neighborhood* $N(v)$ of a vertex v consists of all vertices adjacent to v. A vertex v is *simplicial* when its neighborhood is a clique. The *induced subgraph* $G(U)$ where $U \subseteq V$ is obtained by retaining edges in E where both incident vertices lie in U. When H is an induced subgraph of G, we write $H \subseteq G$. When removing vertices $X \subseteq V$ from G, we denote the resulting induced subgraph $G(V - X)$ as $G - X$. Given an instance M of perfect phylogeny, the *partition intersection graph* (PIG) $G(M)$ has character-state pairs which appear in M as its vertex set, and two vertices are adjacent iff the corresponding character-state pairs appear simultaneously in one of M's rows. Character-state pairs in $G(M)$ are denoted (α_i, r) for character α_i and state r. See Fig. 1.

A graph G is *chordal* if every cycle of length four or more has a *chord* , a pair of vertices that are adjacent in G but not consecutive in the cycle. A *triangulation* of G is a chordal supergraph H of G, and we refer to the edge set $F = E(H) - E(G)$ as a *chordal fill* for G. Given a chordal fill F we use G_F to denote the triangulation of G obtained by adding the edges from F. Triangulations (or chordal fills) are *minimal* if no proper subgraph (subset) is a triangulation (chordal fill). For $G(M)$, we say that a chordal fill F is *legal* iff for

each edge $(\alpha_i, r)(\alpha_j, s) \in F$ we have $i \neq j$. A triangulation H of $G(M)$ is *legal* iff $F = E(H) - E(G(M))$ is a legal chordal fill.

Theorem 1. *[5,16] M has an MD solution iff $G(M)$ has a legal chordal fill.*

Thus, MD reduces to trying to find legal chordal fills for $G(M)$, so we restrict our attention to $G(M)$ and chordal graph theory. Note that in the context of Thm. 1, it suffices to consider minimal chordal fills. Characterizations of chordal graphs and minimal chordal fills are of interest.

Let G be a graph and T a tree. Suppose the nodes x of T are labeled with *bags* $B(x)$ where $B(x) \subseteq V(G)$. When each bag is a maximal clique of G and each induced subgraph $T_v = \{x \in V(T) \mid v \in B(x)\}$ is connected for every $v \in V(G)$, then T is called a *clique tree* for G. Clique trees characterize chordal graphs.

Theorem 2. *[2] A graph G is chordal iff it has a clique tree T.*

Next, suppose that T is a tree and $T_1, \ldots, T_k$ is a collection of subtrees of T. Then the *subtree intersection graph* is the graph with vertex set T_i and edges between two subtrees when they share at least one node in common.

Theorem 3. *[2] A graph G is chordal iff it is isomorphic to the subtree intersection graph for some tree T with subtrees $T_1, \ldots, T_k$.*

For $S \subseteq V$, we say that S is a $uv-$separator if vertices $u, v \notin S$ are not connected in $G - S$. If no proper subset of S is a $uv-$separator then S is a *minimal $uv-$separator* , and S is a *minimal separator* if it is a minimal $uv-$separator for any u and v. This give us our last characterization of chordal graphs.

Theorem 4. *[7] A graph G is chordal iff every minimal separator is a clique.*

A connected component C of $G - S$ is *full* when $N(C) = S$.

Lemma 1. *[14] $S \subseteq V$ is a minimal separator iff it has two or more full components.*

Given two minimal separators S and T, we say that S *crosses* T if S is a $uv-$separator for some $u, v \in T$, writing $S \# T$. This relationship is symmetric [14]. We denote the minimal separators of G by Δ_G. Minimal separators can construct all minimal triangulations in the following way.

Theorem 5. *[14] Every maximal set of pairwise non-crossing minimal separators yields a minimal triangulation by adding edges so that each minimal separator is a clique, and every minimal triangulation is found in this way.*

Minimal separators were used to construct an integer linear program (ILP) in [10] to solve MD. The following lemma will be of use to us.

Lemma 2. *[4] Let x be a vertex of G and $S \in \Delta_{G-x}$. Then either S or $S \cup \{x\}$ is a minimal separator of G.*

For further details, see [6] for an introduction to graph theory, [2] for an introduction to chordal graph theory, and [12] for a survey on minimal triangulations.

3 Character Removal

In this section, we use chordal graph theory to construct an ILP which solves MDCR for an arbitrary number of states.

3.1 Triangulations and Legal Characters

Let M_I be the matrix obtained from M by allowing only columns of M that are indexed via $I \subseteq \{1, \ldots, m\}$. Next we characterize when M_I has a perfect phylogeny in order to solve MDCR.

Given an illegal chordal fill F of $G(M)$, we will construct a set I so that M_I has a perfect phylogeny. Note that $G(M_I)$ is an induced subgraph of $G(M)$. Let I be the set of columns j from M where no fill edges e from F are of the form $e = (\alpha_j, r)(\alpha_j, s)$. We call I the *legal characters* of $G(M)_F$.

Lemma 3. *Let $G(M)_F$ be a minimal triangulation of the partition intersection graph with legal characters I. Then $F(I) = \{(\alpha_i, r)(\alpha_j, s) \in F \mid i, j \in I$ and $i \neq j\}$ is a legal chordal fill for $G(M_I)$.*

Proof. $G(M_I)_{F(I)}$ is an induced subgraph of $G(M)_F$, so any cycle in $G(M_I)_{F(I)}$ is a cycle in $G(M)_F$. Since $G(M)_F$ is chordal, $F(I)$ is a legal chordal fill. □

Lemma 4. *Suppose that $G(M_I)$ has a legal triangulation $G(M_I)_F$ with clique tree T and legal characters I. Then there is a minimal triangulation H of $G(M)$ with legal characters J where $I \subseteq J$.*

Proof. Let T be a clique tree for $G(M_I)_F$ with bags $B(x)$. Let $B'(x) = B(x) \cup \{(\alpha_i, r)(\alpha_i, s) \mid i \notin I$ and $r \neq s\}$. Let H be the unique graph with vertices $V(G(M))$ that is isomorphic to the subtree intersection graph of T and the subtrees $T'_{(\alpha_i, r)} = \{x \in V(T) \mid (\alpha_i, r) \in B'(x)\}$. Thus H is a chordal supergraph of $G(M)$ with legal characters I, and every minimal triangulation H' where $G(M) \subseteq H' \subseteq H$ has at least I as its legal characters. □

Combining Lemmas 3 and 4 with Thm. 1 shows that all submatrices with MD solutions may be found from illegal triangulations. See Fig. 2.

Theorem 6. *M_I has a perfect phylogeny iff some minimal (possibly illegal) triangulation $G(M)_F$ has legal characters J where $I \subseteq J$.*

3.2 An ILP for MDCR

Here, we describe an ILP to solve MDCR. From Theorem 6, the solution to MDCR is the largest submatrix M_I where I is a legal character set of some minimal triangulation of $G(M)$. We use the minimal separators of $G(M)$ to analyze every minimal triangulation via Thm. 5.

Our formulation has variables $x_i \in \{0, 1\}$ for every minimal separator $S_i \in \Delta$ to denote if we add fill edges to turn S_i into a clique. We also have variables

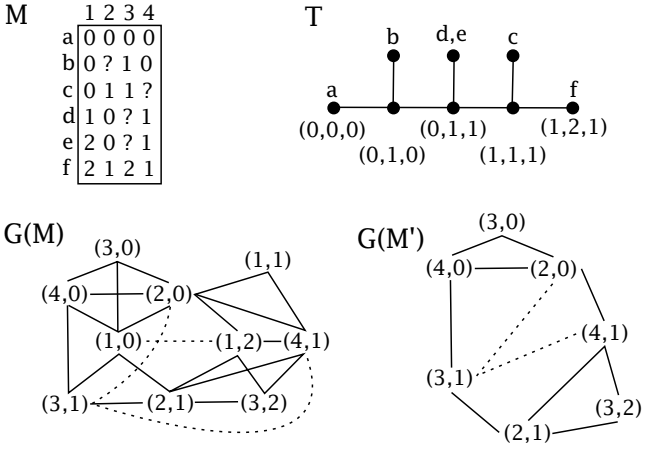


Fig. 2. An instance M of PP with no perfect phylogeny. Here $M' = M_{\{2,3,4\}}$. $G(M)$ has a chordless cycle of length four that alternates between characters 1 and 2, so no perfect phylogeny for M exists. Fill edges are dashed; the illegal triangulation of $G(M)$ induces a legal triangulation of $G(M')$. T is a perfect phylogeny for M' and is an optimal solution to MDCR.

$y_j \in \{0, 1\}$ for each column of M, which will represent the legal characters of a minimal triangulation. Then we wish to maximize

$$\sum_{j=1}^{n} y_j \ . \tag{1}$$

To ensure that no pair of crossing minimal separators $S_i \# S_j$ are chosen, we add the constraint $x_i + x_j \leq 1$. Each minimal separator also requires the maximality constraint

$$x_i + \sum_{j \text{ s.t. } S_j \# S_i} x_j \geq 1 \ . \tag{2}$$

This constraint makes sure that feasible solutions either pick S_i or some minimal separator that crosses it. Lastly, for every column j with two character-state pairs in S_i we use the constraint $x_i \leq (1 - y_j)$ so that the legal characters are properly calculated.

In both our solution to MDCR and the solution to MD in [10], a critical step is to calculate the minimal separators of $G(M)$. Our formulation can be easily extended to the weighted case by modifying the objective function. In theory, there are worst case $O(2^{|V|})$ minimal separators which are calculated in $O(|V|^3|\Delta_G|)$ time [1]. This motivates our preprocessing technique in the next section which is used to find the minimal separators of $G(M)$ faster.

4 Preprocessing $G(M)$

Here, we study the minimal separators of a graph after removing simplicial vertices. We will see that doing so removes only clique minimal separators, resulting in a subgraph that we can use to find any minimal fill for G.

Theorem 7. *Suppose that X and Y are vertices from a graph G that are obtained by greedily removing simplicial vertices, and that $G - X$ and $G - Y$ have no simplicial vertices. Then $X = Y$.*

When X is as in Thm. 7, we call $\mathcal{S}(G) = G - X$ the *separator core* of G.

Lemma 5. *Let x be a simplicial vertex. Then for each $S \in \Delta_G$ either $S \in \Delta_{G-x}$ or S is a clique with two full components, and one of them is $\{x\}$.*

Proof. Simplicial vertices are never in any minimal separator, so there is some connected component $C \in \mathcal{C}_G(S)$ where $x \in C$. Removing x from G will only affect C, so that all other connected components of S remain intact. If $G - S$ has two full components which are not C, then we must have $S \in \Delta_{G-x}$. Thus it suffices to consider when C is one of two full components. Suppose $C \neq \{x\}$. $C - \{x\}$ is still full as x is simplicial, since every neighbor $y \in N(x)$ is adjacent to each vertex in $N(x) \cap N(C)$ so $S \in \Delta_{G-x}$. Lastly, assume $C = \{x\}$. Since C is full, $S = N(C) = N(x)$ is a clique since x is simplicial. Further, $\mathcal{C}_{G-x}(S)$ has only one full component and is no longer a minimal separator, completing the proof. □

That is, trimming away simplicial vertices only destroys clique minimal separators. Moreover, minimal separators are never created by removing simplicial vertices, as we see next.

Lemma 6. *Let $S \in \Delta_{G-x}$ with x simplicial. Then $S \in \Delta_G$.*

Proof. By Lemma 2, either S or $S \cup \{x\}$ is a minimal separator of G. But simplicial vertices are never in any minimal separator so we must have $S \in \Delta_G$. □

Thus we have that $\Delta_{G-x} \subseteq \Delta_G$, and by Lemma 5 if $S \in \Delta_G - \Delta_{G-x}$ it is a clique. No clique minimal separator can cross another minimal separator. Along with Theorem 5, this yields the following lemma.

Lemma 7. *Let $\Phi \subseteq \Delta_G$ be a maximal pairwise noncrossing set of nonclique minimal separators of G. Then G_Φ is a minimal chordal fill, and every minimal chordal fill can be obtained in this way.*

By Lem. 5 and Thm. 4 we only destroy clique minimal separators when trimming simplicial vertices. Combining this with Lem. 7, it suffices to consider the separator core when computing minimal fills.

Theorem 8. *Let $\Phi \subseteq \Delta_{\mathcal{S}(G)}$ be a maximal pairwise noncrossing set of minimal separators of the separator core of G. Then G_Φ is a minimal triangulation, and every minimal triangulation of G is obtained in this way.*

Table 1. 80 × 80 Instances of PP

p	r	v	e	Δ	K	C_1	C_2	S_1^C	S_2^C	Tot$_1$	Tot$_2$	S_1	S_2
10 Maxstates													
0.0	0.0	0.32	0.25	0.33	0.00	2.19	2.21	1.46	0.77	3.65	3.09	12.57	11.18
0.05	0.0	0.40	0.33	0.41	0.00	2.05	2.08	2.37	1.38	4.42	3.59	13.58	11.64
0.1	0.0	0.47	0.41	0.46	0.00	1.90	1.92	3.16	2.20	5.06	4.04	14.07	12.39
0.15	0.0	0.57	0.52	0.58	0.00	1.75	1.77	5.53	3.50	7.28	5.26	16.15	13.34
0.2	0.0	0.63	0.60	0.66	0.02	1.62	1.62	11.17	5.80	12.79	7.47	21.45	14.79
0.0	0.05	0.34	0.27	0.42	0.00	2.15	2.12	3.00	0.94	5.15	3.09	16.04	10.75
0.05	0.05	0.40	0.33	0.49	0.00	2.01	2.00	4.40	1.21	6.42	3.26	17.87	10.91
0.1	0.05	0.50	0.44	0.63	0.02	1.88	1.85	7.25	2.37	9.13	4.24	21.02	12.38
0.15	0.05	0.59	0.55	0.73	0.06	1.75	1.69	12.15	4.38	13.90	6.06	25.65	13.38
0.2	0.05	0.65	0.62	0.85	0.08	1.68	1.62	17.66	6.63	19.34	8.45	30.74	16.14
0.0	0.75	0.36	0.29	0.44	0.00	2.15	2.15	2.80	1.45	4.95	3.68	15.52	12.71
0.05	0.75	0.41	0.35	0.49	0.00	2.01	1.99	3.93	2.27	5.95	4.27	16.87	13.32
0.1	0.75	0.50	0.44	0.58	0.00	1.88	1.85	6.58	3.12	8.46	5.01	19.82	14.06
0.15	0.75	0.58	0.54	0.63	0.02	1.74	1.71	10.44	4.88	12.19	6.56	23.31	15.16
0.2	0.75	0.65	0.63	0.70	0.04	1.63	1.57	16.47	8.91	18.10	10.67	29.18	20.26
20 Maxstates													
0.0	0.0	0.33	0.26	0.52	0.04	2.92	2.80	10.25	2.62	13.16	5.29	35.76	21.67
0.05	0.0	0.43	0.36	0.61	0.06	2.72	2.61	14.51	4.13	17.24	6.87	38.10	21.98
0.1	0.0	0.53	0.48	0.67	0.10	2.56	2.49	23.17	8.91	25.73	11.35	45.86	27.69
0.15	0.0	0.60	0.56	0.67	0.16	2.47	2.25	31.62	12.83	34.09	15.35	50.82	31.07
0.2	0.0	0.64	0.62	0.68	0.20	2.35	2.15	42.09	23.42	44.44	25.65	60.45	45.28
0.0	0.05	0.35	0.29	0.59	0.12	2.97	2.88	19.93	6.15	22.89	9.41	51.00	29.23
0.05	0.05	0.44	0.38	0.59	0.16	2.84	2.64	28.19	9.66	31.04	12.46	56.75	33.78
0.1	0.05	0.53	0.48	0.63	0.14	2.56	2.45	35.59	18.09	38.15	20.66	60.89	44.55
0.15	0.05	0.59	0.56	0.57	0.26	2.51	2.34	50.03	27.04	52.54	29.55	69.54	54.48
0.2	0.05	0.62	0.61	0.70	0.38	2.28	2.20	57.96	37.34	60.24	39.46	77.92	76.12
0.0	0.75	0.35	0.29	0.61	0.20	3.15	2.95	29.24	4.02	32.39	6.87	65.48	24.16
0.05	0.75	0.43	0.38	0.62	0.22	2.98	2.78	37.10	6.57	40.08	9.71	65.09	26.53
0.1	0.75	0.53	0.49	0.60	0.28	2.69	2.54	45.48	14.18	48.17	16.49	68.28	34.30
0.15	0.75	0.59	0.57	0.47	0.36	2.57	2.41	58.78	20.58	61.35	22.91	74.15	44.90
0.2	0.75	0.64	0.62	0.51	0.36	2.41	2.24	62.14	30.29	64.55	32.59	75.50	54.83

Proof. When $\Phi \subseteq \Delta_G$, we use $F(\Phi)$ to denote the fill edges required to make each $S \in \Phi$ a clique.

Suppose Φ is as above. Then if $S \in \Delta_G - \Phi$ it is either a clique or crosses some $T \in \Phi$. Letting Δ_C denote the clique minimal separators of G, $\Phi \cup \Delta_C$ is chordal by Thm. 5 and $F(\Phi) = F(\Phi \cup \Delta_C)$. Conversely, let Φ be as in Thm. 5 and $\Phi' \subseteq \Phi$ the minimal separators of Φ which are not complete in G. Then $F(\Phi) = F(\Phi')$ and $\Phi' \subseteq \Delta_{S(G)}$, completing the proof. □

Thus we can use $\Delta_{S(G)}$ to compute minimal triangulations of G. Our implementation utilizes the *deficiency set* $D(v)$ of a vertex v, where $D(v) = \{uw \mid u, w \in N(v) \text{ and } uw \notin E\}$, along with its inverse $D^{-1}(e) = \{v \mid e \in D(v)\}$. We keep track of each inverse deficiency set along with the size of each deficiency set. When a vertex has an empty deficiency set it is simplicial and removed. For each edge e incident to a simplicial vertex, we use $D^{-1}(e)$ to efficiently update the size of the deficiency sets and repeat this process until no simplicial vertices remain.

5 Empirical Results

Our preliminary MDCR tests were performed on 4 state (DNA) data, with matrix sizes 80 by 245, 41 by 1043, and 50 by 176. In each instance, computation completed and a solution to MDCR was found.

Next we present preliminary empirical results for the separator core as a precomputing tool to calculate minimal separators for partition intersection graphs. Tests were run in emulated Ubuntu 8.04 (via VMWare Player) with a 2.4 ghz dual core processor that was allocated 2 gigabytes of RAM.

Table 1 corresponds to a square matrix with dimensions as specified. Each row shows the results of 50 runs. Data for each run was generated by the coalescent based program ms [13]. This approach mirrors the approach for generating data in [10]. p is the average number of missing entries, r is a coalescent parameter used by ms, v (e) is the number of core vertices (edges) divided by the number of vertices (edges) in the original graph, Δ is the number of minimal separators of the core graph divided by the number of minimal separators of the original graph (when this computation finished), K is the percent of processes which were killed by the OS. The next four pairs of columns contain averages and medians, where C_i denotes core computation time, S_i^C denotes minimal separator computation time for the core, Tot_i denotes total core and minimal separator computation time, and S_i denotes minimal separator computation time for the original graph.

We see consistent reductions in the amount of vertices, edges, and minimal separators reduced. In moderate sized instances, we see that the time is reduced to roughly a third to a half of the original time. More tables may be found at `http://wwwcsif.cs.ucdavis.edu/~gyselr/CR_Core_ISBRA10`

6 Conclusions

We used triangulations of the partition intersection graph to formulate an ILP solution to MDCR. Chordal graphs and minimal triangulations have many characterizations [2,12] that may lead to other ILP formulations that take advantage of Thm. 6.

We also outlined a useful preprocessing tool for the calculation of minimal separators. Intuitively, our approach seems useful for partition intersection graphs as they are a collection of overlaying cliques and seem to have small separator cores. A naive implementation of our algorithm takes $O(n^3)$ time, and it would be of interest to find a better algorithm.

Acknowledgments

We thank Fumei Lam for programming and testing the ILP formulation for our approach to MDCR in Sect. 3.2. We thank the anonymous reviewers and Balaji Venkatachalam for careful reading and helpful comments. This research was partially supported by NSF grants SEI-BIO 0513910, CCF-0513910, and IIS-0803564.

References

1. Berry, A., Bordat, J.P., Cogis, O.: Generating all the minimal separators of a graph. International Journal of Foundations of Computer Science 11, 397–403 (2000)
2. Blair, J.R.S., Peyton, B.W.: An introduction to chordal graphs and clique trees. Institute for Mathematics and Its Applications 56, 1–30 (1993)
3. Bodlaender, H., Fellows, M., Warnow, T.: Two strikes against perfect phylogeny. In: Proc. of the 19th Inter. colloquium on Automata, Languages and Programming, pp. 273–283 (1992)
4. Bouchitte, V., Todinca, I.: Listing all potential maximal cliques of a graph. Theoretical Computer Science 276, 17–32 (2002)
5. Buneman, P.: A characterization of rigid circuit graphs. Discrete Mathematics 9, 205–212 (1974)
6. Diestel, R.: Graduate texts in mathematics 173: Graph Theory. Springer, Heidelberg (2000)
7. Dirac, G.A.: On rigid circuit graphs. Abh. Math. Sem. Univ. Hamburg 25, 71–76 (1961)
8. Felsenstein, J.: Inferring Phylogenies. Sinauer, Sunderland (2004)
9. Fernandez-Baca, D.: The perfect phylogeny problem. In: Du, D.Z., Cheng, X. (eds.) Steiner Trees in Industries. Kluwer Academic Publishers, Dordrecht (2000)
10. Gusfield, D.: The multi-state perfect phylogeny problem with missing and removable data: solutions via integer-programming and chordal graph theory. In: Batzoglou, S. (ed.) RECOMB 2009. LNCS, vol. 5541, pp. 236–252. Springer, Heidelberg (2009)
11. Gusfield, D., Frid, Y., Brown, D.: Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data. In: Lin, G. (ed.) COCOON 2007. LNCS, vol. 4598, pp. 51–64. Springer, Heidelberg (2007)
12. Heggernes, P.: Minimal triangulation of graphs: a survey. Discrete Mathematics 306(3), 297–317 (2006)
13. Hudson, R.: Generating samples under the Wright-Fisher neutral model of genetic variation. Bioinformatics 18(2), 337–338 (2002)
14. Parra, A., Scheffler, P.: Characterizations and algorithmic applications of chordal graph embeddings. Discrete Applied Mathematics 79, 171–188 (1997)
15. Steel, M.: The complexity of reconstructing trees from qualitative characters and subtrees. J. of Classification 9, 91–116 (1992)
16. Semple, C., Steel, M.A.: Phylogenetics. Oxford University Press, UK (2003)

Analysis of Gene Interactions Using Restricted Boolean Networks and Time-Series Data

Carlos H.A. Higa, Vitor H.P. Louzada, and Ronaldo F. Hashimoto

University of São Paulo, São Paulo, SP, Brazil
{higa,louzada,ronaldo}@ime.usp.br

Abstract. A popular model for gene regulatory networks is the Boolean network model. In this paper, we propose an algorithm to perform an analysis of gene regulatory interactions using the Boolean network model and time-series data. Actually, the Boolean network is restricted in the sense that only a subset of all possible Boolean functions are considered. We explore some mathematical properties of the restricted Boolean networks in order to avoid the full search approach. We applied the proposed algorithm in a case study of the budding yeast cell cycle network using an artificial dataset. The results show that some interactions can be fully or, at least, partially determined under the Boolean model considered. We have shown that this analysis can be used as the first step for gene relationships detection with a high flexibility to include biological knowledge. What we envisage with our method is a model that points out which connections should be checked in the wet lab and consequently facilitate some biological experiments.

1 Introduction

Some of the goals of Systems Biology is to study the various cellular mechanisms and components. In many cases, these mechanisms are complex, where some of the interactions between the proteins are still unknown. To represent these interactions it is common to use gene regulatory networks (GRN). There are several models of GRN, from discrete to continuous models. The simplest discrete model was introduced by Kauffman [1] and its known as *Boolean network* model. Later, this model was modified to express uncertainty giving rise to the *probabilistic Boolean network* model [2,3]. Friedman introduced *Bayesian networks* [4] as a probabilistic tool for the identification of regulatory data and showed that they can reproduce certain known regulatory relationships. Among the continuous models we can cite the *ordinary differential equations* model which was suggested several decades ago [5]. For a more detailed review about models of gene regulatory networks see [6].

Models of gene regulatory networks help us to study biological phenomena (e.g. cell cycle) and diseases (e.g. cancer). Therefore, unreaveling such networks, or at least some of its connections, is an important problem to address. The ability to uncover the mechanisms of GRN has been possible due to developments in high-throughput technologies, allowing scientists to perform analysis

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 61–76, 2010.

on the DNA and RNA levels. The most common type of data generated by these technologies are gene expression data (microarray).

The biological systems are notoriously complex. Determining how the pieces of this puzzle come together to create living systems is a hard challenge known as *reverse engineering*, which is the process of elucidating the structure of a system by reasoning backwards from observations of its behavior [7]. GRN in many cases cannot be unraveled precisely, however, because of measurement noise and the limited number of data sets compared with the number of genes that are involved.

The most common approach to reverse engineering GRN is to use gene expression data. Some algorithms use additional information from heterogeneous data sources, e.g. genome sequence and protein-DNA interaction data, to assist the inference process. Hecker et al. [8] presents a good review about GRN inference and data integration.

Usually, an inference algorithm aims to construct one single network which is believed to be the real network. The issue is that the inverse problem is ill-posed, meaning that several networks could explain (or generate) the data set given as the input for the algorithm. The problem becomes more complicated if we take into account the noise that may be present in the data and the small amount of samples. For this reason, our approach aims to analyze the network in a statistical manner. Our algorithm creates several networks that could explain the data. By analyzing the similarities among these networks, we will propose a confidence measure of the regulatory relationship between the genes.

In this paper, we present an algorithm based on Boolean networks and time-series gene expression. Actually, the Boolean networks are called *restricted* in the sense that not all Boolean functions are allowed in the model. Restricting the network reduces the search space, which can be significant given that the inverse problem is very complex. The time-series data allow us to observe part of the dynamics of the system. These observations are used to infer the regulatory relationships between the genes.

A challenge always presented in any gene regulatory model is its usefulness. It would be interesting if a model could help biological experiments in understanding gene interactions. The model here presented is capable of inferring some of these connections from time-series data of gene expressions, and this inference process is helped by all *a priori* knowledge available. What we envisage with our method is a model that points out which connections should be determined in the wet lab that would constrain as many other connections as possible and consequently could facilitate some biological experiments.

The paper is organized as follows. In the next section we present the restricted Boolean network model. The algorithm for the statistical analysis is presented in Sect. 3. A budding yeast cell-cycle model from which the artificial data are obtained is described in Sect. 4. In Sect. 5 and 6 we show and discuss our results and we conclude the work in Sect. 7.

2 Restricted Boolean Network Model

A Boolean network (BN) is defined by a set $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$ of n Boolean variables and a set $\mathbf{F} = \{f_1, f_2, \ldots, f_n\}$ of n Boolean functions. In the case of GRN the variables are called genes. Obviously, each gene x_i, $i = 1, \ldots, n$, can assume only two possible values: 0 (OFF) or 1 (ON). The value of the gene x_i at time $t+1$ is determined by genes $x_{j_1(i)}, x_{j_2(i)}, \ldots, x_{j_{k_i}(i)}$ at time t through a Boolean function $f_i : \{0,1\}^{k_i} \rightarrow \{0,1\}$. Given that, there are k_i genes assigned to gene x_i, and the mapping $j_k : \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}$, $k = 1, \ldots, k_i$ determines the "wiring" of x_i [9]. This way,

$$x_i(t+1) = f_i(x_{j_1(i)}(t), x_{j_2(i)}(t), \ldots, x_{j_{k_i}(i)}(t)) \ . \tag{1}$$

We assume that all genes are updated synchronously by the functions in $\mathbf{F}$ assigned to them and this process is repeated. The artificial synchrony simplifies computation while preserving the qualitative, generic properties of global network dynamics [11,10]. A *state* of the network at time t is a binary vector $s(t) = (x_1(t), \ldots, x_n(t))$. Therefore, the number of states is 2^n, labeled by $s_0, s_1, \ldots, s_{2^n-1}$. The dynamics of the network is represented by the transition between states. This model is deterministic given that there is a single Boolean function to regulate each gene. Because of the finite number of states and the deterministic behavior, some of the states may be visited cyclically. These states form what is known by the *attractor* of the BN. The states outside the attractor are called *transient* states. The transient states together with the corresponding attractor states forms the *basin of attraction* of that attractor.

In the case of restricted Boolean networks, the regulatory relationships is represented by a matrix $A_{n \times n}$ using the following convention: $a_{ij} = 1$ for a positive regulation from gene x_j to gene x_i; $a_{ij} = -1$ for a negative regulation from x_j to x_i; For the remaining cases $a_{ij} = 0$. The Boolean function f_i is defined according to the matrix A and the values of the genes x_j, $j = 1, \ldots, n$, at time t:

$$x_i(t+1) = \begin{cases} 1, & \text{if } \sum_j a_{ij} x_j(t) > 0 \\ 0, & \text{if } \sum_j a_{ij} x_j(t) < 0 \\ x_i(t), & \text{if } \sum_j a_{ij} x_j(t) = 0 \ . \end{cases} \tag{2}$$

We call the summation $\sum_j a_{ij} x_j(t)$ the *input* of x_i at time t. Besides the regulatory relationships of the matrix A, each gene can have a self-degradation behavior. A gene x_i with self-degradation is set to 0 whenever its input is null. Observe that not all Boolean functions can be represented using (2) and that is why the Boolean network is called "restricted". In Sect. 4, we will present a budding yeast cell-cycle model proposed by Li et al. [14] which is based on restricted Boolean networks. This model will be used to perform the statistical analysis algorithm.

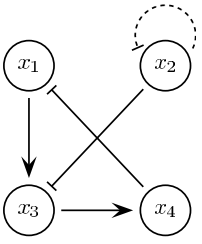


Fig. 1. Small example containing four genes

Example: Let us show a small example of a restricted Boolean network containing only four genes. Fig. 1 shows the regulatory relationship between the four genes. An arrow is a positive regulation; a line with a bar at the end is a negative regulation; the dotted loop on x_2 indicates that this gene has a self-degradation behavior.

Given the regulatory relationships in Fig. 1, the corresponding regulation matrix is presented below:

$$A = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array} . \tag{3}$$

Applying the Boolean function given by (2) for every possible state, we can construct a state transition diagram, shown in Fig. 2. As we can see, there are three attractors: 0000, 0001 and 0011; the remaining states are transient states. The attractor 0011 has the largest basin of attraction (we consider the number of states as the size of the basin of attraction).

3 Gene Interaction Analysis Algorithm

The algorithm was designed under the assumption that the gene expression data were generated by a biological system which can be modeled as a restricted Boolean network. Let $\mathbf{S} = \{S(1), S(2), \ldots, S(m)\}$ be a set of m time-series gene expression profiles, where $S(i) \in \{0,1\}^n$ for $i = 1, \ldots, m$. The algorithm aims to analyze networks that produce the sequence

$$S(1) \rightarrow S(2) \rightarrow \cdots \rightarrow S(m) \ . \tag{4}$$

When the network produces the time-series data we say that the network is *consistent* with the data. Naturally, there may exist several consistent networks for a single sequence. That is, the inverse problem is a "one-to-many" or ill-posed problem, and this is very difficult to handle.

One naïve way to solve this ill-posed problem is to find all possible networks by a full search algorithm. In fact, Lau et al. [12] proposed a "smart" full search

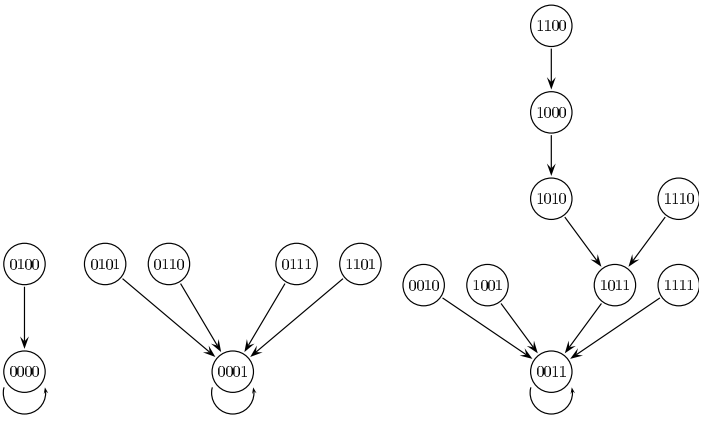


Fig. 2. State transition diagram of the restricted Boolean network shown in Fig. 1

algorithm to enumerate all possible networks. Here, in this paper, we explore some mathematical properties of the restricted Boolean networks in order to avoid this full search approach.

The algorithm uses an encoding to represent the interaction between a pair of genes. Table 1 shows the code and its respective subset of possible interactions where -1, 0 and 1 stand for *inhibition*, *no relationship* and *activation*, respectively. At the beginning of the process, the relationships between genes are unknown and they are represented by a matrix $A_{n\times n}$ filled with the code 5. This means that any edge (activation or repression) or none can occur (the regulatory relationship is *undetermined*). As the process runs, the entries of the matrix can change to -2, 2 or 3 (*partially determined* relation). In addition, if an entry of the matrix is *completely determined* we can set its value to -1, 0 or 1. At the end of the process the entries of A can hold undetermined, partially determined or determined values. The undetermined and partially determined entries can lead to several matrices that represent a consistent network.

3.1 The Three Steps of the Algorithm

The algorithm aims to uncover the hidden relationships between the genes through the information provided by the time-series sequence, which can be seen as a state transition sequence of the corresponding BN. The algorithm consists in three main steps applied cyclically. Next, we will explain the concepts used in each step.

Step one. The first step of the algorithm analyzes the sample in triplets, $S(t-1)$, $S(t)$ and $S(t+1)$. An important point to notice here is that if two consecutive states $S(t-1)$ and $S(t)$ differ only in one single gene x_k, then any gene x_i that had its value changed from $S(t)$ to $S(t+1)$ is directly regulated by x_k. To illustrate this situation, consider the time-series data (Table 2) extracted from the example given in Sect. 2. Looking at the time points $S(1)$ and $S(2)$ we

Table 1. Encoding table used in the algorithm

Code	Subset
-1	$\{-1\}$
0	$\{0\}$
1	$\{1\}$
-2	$\{-1, 0\}$
2	$\{0, 1\}$
3	$\{-1, 1\}$
5	$\{-1, 0, 1\}$

observe that only x_2 had its value changed (from 1 to 0). Now, looking at $S(2)$ and $S(3)$ we can see that x_3 was turned to 1. Following the restricted Boolean network model, this change was caused, necessarily, by the gene x_2. In fact, x_2 inhibits x_3 at time $t = 1$ and it is self degraded at time $t = 2$, allowing x_1 to activate x_3 at time $t = 3$. Using this approach, we state the following proposition (Proposition 1).

Table 2. Time-series data taken from Fig. 2

t	$x_1(t)$	$x_2(t)$	$x_3(t)$	$x_4(t)$
1	1	1	0	0
2	1	0	0	0
3	1	0	1	0
4	1	0	1	1
5	0	0	1	1

Proposition 1. *Let $S(t-1)$, $S(t)$ and $S(t+1)$ be three consecutive states according to the restricted Boolean network model. If $S(t-1)$ and $S(t)$ differ by a single gene x_k, then for each gene x_i such that $x_i(t) \neq x_i(t+1)$ we have that x_k regulates x_i directly, that is, $a_{ik} \neq 0$.*

Proof. Suppose that $S(t-1)$ and $S(t)$ differ by a single gene x_k, and that there is at least one gene x_i such that $x_i(t) \neq x_i(t+1)$. As $x_i(t) \neq x_i(t+1)$, the summations $\sum_j a_{ij}x_j(t-1)$ and $\sum_j a_{ij}x_j(t)$ have different signs. Given that x_k is the only gene possessing different values in $S(t-1)$ and $S(t)$, this difference signal must have been caused by x_k. Therefore, $a_{ik} \neq 0$.

The type of the regulatory relationship (activation or inhibition) uncovered using Proposition 1 depends on the values of x_k and x_i. Table 3 lists all possible combinations of values for x_k (time $t-1$ and t) and x_i (time t and $t+1$). We call these relationships as *required*, since they must be present in the network in order to maintain the consistency with the time-series data.

The approach used in Proposition 1 can be extended when $S(t-1)$ and $S(t)$ differ in more than one gene. For example, let us say that two genes, x_{k_1} and

Table 3. All possible combinations of values for x_k and x_i

$x_k(t-1)$	$x_k(t)$	$x_i(t)$	$x_i(t+1)$	type
0	1	0	1	activation
0	1	1	0	inhibition
1	0	0	1	inhibition
1	0	1	0	activation

x_{k_2}, are the genes differently expressed from $S(t-1)$ to $S(t)$. If x_i had its value changed from $S(t)$ to $S(t+1)$ there are some *regulation hypotheses* that we must take into account. Analyzing x_{k_1} and x_{k_2} individually, we can use the Table 3 to generate two hypotheses and, these two hypotheses must be combined to generate a third hypothesis. For instance, we can infer that x_{k_1} activates x_i and x_{k_2} inhibits x_i, not in the same network. Given that, a third network would consider both hypotheses simultaneously. This way, the number of hypotheses grows in a combinatorial manner.

Step Two. The second step of the algorithm takes into account two consecutive states, $S(t)$ and $S(t+1)$. There is one important observation here: only the active genes at time t can possibly regulate genes at time $t+1$. This fact becomes clear when we look at (2). The active genes can give us an insight of which genes are regulating other gene, although the type of the regulatory relationship can not be determined. However, the input given by the summation in (2) can help us to determine the regulatory relationships. For example, if we observe that a gene x_i changes its value from 0 (at time t) to 1 (at time $t+1$), we can deduce that the input for gene x_i is positive at time t and only the active genes at time t are responsible for this positive input. Following this logic, the algorithm generates all possible combinations of regulatory relationships using the active genes such that the input of gene x_i at time t is coherent to the values of x_i at time $t+1$.

To exemplify, consider the data in Table 2 where $t = 3$. At this time, there are two active genes, x_1 and x_3. These genes are the only ones that can contribute to the sign of the input for each gene. If we look at the gene x_4 we observe that its value turned from 0 to 1. According to (2), the input must be positive in this case, that is, $\sum_{j=1}^{4} a_{4j}x_j(3) > 0$. Given that, we must have $a_{41} + a_{43} > 0$. Therefore, neither a_{41} or a_{43} can take the value -1, only 1 or 0 (not both). The same logic can be applied to all genes and then, the information extracted using this approach can support the inference procedure.

Step Three. The third step analyzes any two pairs of consecutive states in the time-series data. Let t_1 and t_2 be two time points in the time-series data:

$$S(1) \rightarrow \cdots \rightarrow S(t_1) \rightarrow S(t_1+1) \rightarrow \cdots \rightarrow S(t_2) \rightarrow S(t_2+1) \rightarrow \cdots \rightarrow S(m) \ . \tag{5}$$

Now, let us suppose that $S(t_1)$ and $S(t_2)$ are very similar. Hence, the difference between $S(t_1+1)$ and $S(t_2+1)$ must be caused by the differentially expressed

genes of their predecessors. For instance, let us suppose that $S(t_1)$ and $S(t_2)$ differ in one single gene:

$$S(t_1) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \ , \ S(t_2) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \ . \tag{6}$$

And the succession occurs as stated:

$$\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \ , \cdots \ , \ \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \longrightarrow \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \ . \tag{7}$$

Therefore, the huge difference between $S(t_1+1)$ and $S(t_2+1)$ in this case must be caused by the change on x_4. In this step, the algorithm checks how each gene changed in the two pairs of consecutive states.

In our example, let us concentrate on gene x_1. It was inhibited in the first pair and had no change in the second pair. Let I be the total input originated in the genes with similar expression in $S(t_1)$ and $S(t_2)$, M be the input generated by x_4 in $S(t_1)$, and $\bar{M}$ be the input generated by x_4 in $S(t_2)$. Therefore, to explain the changes of x_1 in the two pairs, we must have:

$$\begin{cases} I + M < 0 \text{ and} \\ I + \bar{M} \geq 0 \ . \end{cases} \tag{8}$$

If a_{ij} represents the influence of gene x_j over x_i, we can calculate I, M and $\bar{M}$ as follows:

$$I = \begin{pmatrix} a_{11} & a_{12} & a_{13} \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = a_{11} + a_{13} \ , \tag{9}$$

$$M = a_{14} \cdot 0 = 0 \text{ and} \tag{10}$$

$$\bar{M} = a_{14} \cdot 1 = a_{14} \ . \tag{11}$$

Henceforth,

$$\begin{cases} I + M < 0 \\ I + \bar{M} \geq 0 \end{cases} \implies \begin{cases} a_{11} + a_{13} + \ 0 \ < 0 \\ a_{11} + a_{13} + a_{14} \geq 0 \end{cases} \implies \begin{cases} a_{14} > 0 \ . \end{cases} \tag{12}$$

This result implies that the entry a_{14} of the matrix must have the code 1.

If $S(t_1)$ and $S(t_2)$ differ in more than one gene, we can still generate hypotheses of regulation. In fact, this step tries to construct a system of inequalities with the inputs of each gene for every combination of two consecutive pairs.

3.2 Analysis of Gene Interactions

The three steps of the algorithm are performed cyclically until no additional information can be included in the matrix A. At this point, the entries of A are filled with the regulatory hypotheses generated by the algorithm. Some of the entries represent the undetermined or partially determined relationships between genes.

We can think of A as a root of a tree where the leaves are the matrices that can be generated from the root by determining a value for each partially determined/undetermined entry. Perhaps, this value determination can be guided by biological knowledge. In Fig. 3 we show an example using four genes as presented in Sect. 2. There are two partially determined entries in the root (marked with bold face numbers) that can be determined one at a time, generating four possible matrices in the second level of the tree. After determining an entry, the three steps of the algorithm are performed again as previously and the overall process is repeated until a completed determined matrix (a leaf of the tree) is obtained. Some of the leaves are consistent matrices, that is, they represent a network consistent with the data.

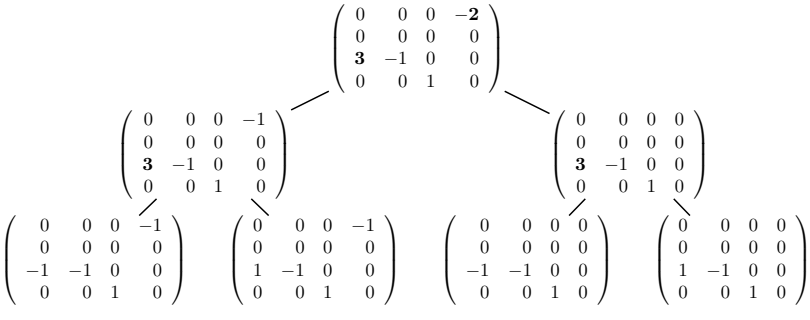


Fig. 3. The root of the tree and the possible matrices generated from the root

In order to analyze the gene interactions, since there may be a combinatorial explosion in generating the matrices, we randomly generate some of them (a sampling process) and consider the consistent ones to perform the analysis. In the worst case, this algorithm has exponential running time. However, it does not generates all the 3^{n^2} possible matrices. In the next section, we present a Boolean model of the budding yeast cell cycle that was used to generate artificial data to apply the algorithm and, in Sect. 5, we show the results.

4 Budding Yeast Cell Cycle Model

The cell-cycle process consists of four phases: G_1 (in which the cell grows and, under appropriate conditions, commits to division), S (in which the DNA is synthesized and chromosomes replicated), G_2 (a"gap" between S and M), and M (in which chromosomes are separated and the cell is divided in two). After

the M phase, the cell returns to the G_1 phase, waiting for appropriate conditions for another round of division. We call this G_1 phase as stationary G_1. There are ≈ 800 genes involved in the cell-cycle process of the budding yeast [13]. However, the number of key regulators that are responsible for the control and regulation of this complex process is much smaller [14].

The budding yeast cell-cycle model proposed by Li et al. [14] is based on a network of eleven regulators, as shown in Fig. 4. The meaning of the edges are the same as in Fig. 1. The eleven genes $x_1, \ldots, x_{11}$ are Cln3, MBF, SBF, Cln1, Cdh1, Swi5, Cdc20, Clb5, Sic1, Clb1, and Mcm1, respectively. The "cell-size" node was introduced just to indicate a checkpoint to start the cell-cycle process.

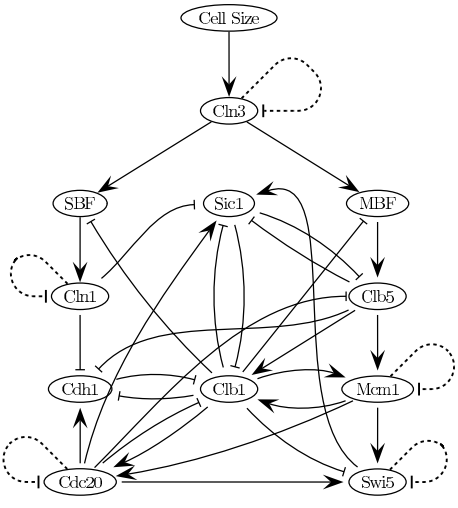


Fig. 4. The cell cycle network of the budding yeast

Considering the restricted Boolean network model presented in Sect. 2, Li et al. [14] studied the dynamics of the network. They found that there are seven attractors, shown in Table 4. In this table, each row represents an attractor where the first column indicates the size of the basin of attraction. There is one big basin composed by 1,764 or $\approx 86\%$ of states. According to Li et al. [14], the corresponding attractor is the biological G_1 stationary state.

Biologically, the cell-cycle sequence starts when the cell commits to division by activating Cln3. To simulate the cell cycle, they started the process by "exciting" the G_1 stationary state with the cell size signal, that is, inducing the gene Cln3 to an active state. Applying (2) to simulate the process it was observed that the system goes back to the G_1 stationary state. The temporal evolution of the states, presented in Table 5, follows the cell-cycle sequence, going from excited G_1 state (Start) to the S phase, the G_2 phase, the M phase, and finally to the stationary G_1 state. This is the biological trajectory or pathway of the cell-cycle network.

Table 4. The seven attractors of the cell-cycle network

Basin size	Cln3	MBF	SBF	Cln1	Cdh1	Swi5	Cdc20	Clb5	Sic1	Clb1	Mcm1
1,764	0	0	0	0	1	0	0	0	1	0	0
151	0	0	1	1	0	0	0	0	0	0	0
109	0	1	0	0	1	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0	1	0	0
7	0	1	0	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0

Table 5. Temporal evolution of states for the cell-cycle network

Time	Cln3	MBF	SBF	Cln1	Cdh1	Swi5	Cdc20	Clb5	Sic1	Clb1	Mcm1	Phase
1	1	0	0	0	1	0	0	0	1	0	0	Start
2	0	1	1	0	1	0	0	0	1	0	0	G_1
3	0	1	1	1	1	0	0	0	1	0	0	G_1
4	0	1	1	1	0	0	0	0	0	0	0	G_1
5	0	1	1	1	0	0	0	1	0	0	0	S
6	0	1	1	1	0	0	0	1	0	1	1	G_2
7	0	0	0	1	0	0	1	1	0	1	1	M
8	0	0	0	0	0	1	1	0	0	1	1	M
9	0	0	0	0	0	1	1	0	1	1	1	M
10	0	0	0	0	0	1	1	0	1	0	1	M
11	0	0	0	0	1	1	1	0	1	0	0	M
12	0	0	0	0	1	1	0	0	1	0	0	G_1
13	0	0	0	0	1	0	0	0	1	0	0	Stationary G_1

The states presented in Table 5 are used as the time-series data to perform the statistical analysis. The results are shown in the next section.

5 Results

The application of the algorithm presented in Sect. 3 creates a collection of consistent networks totally inferred from the time-series data of the yeast cell cycle. If we calculate the frequency of the connections, we are capable of assigning probabilities to each gene relationship. In Fig. 5 and 6 we show the frequency of different types of inward connections to each gene from all other genes. Evidently, the determined connections will appear with frequency 100% in all the networks; while the partially determined connections will have, at least, one gene relationship (activation, no connection or inhibition) with frequency 0%.

From the frequencies shown in Fig. 5 and 6, we can see that the algorithm was capable of identifying 11 determined connections and 13 partially determined connections. The results are shown in Fig. 7. Note that, in this figure, the arrows do not indicate activation necessarily.

6 Discussion

By looking at Fig. 5 and 6, it is interesting to note that, in some cases, the statistics of the networks were capable of almost excluding one possibility of relationship - as shown in Swi5 → Cln3, SBF → Clb5, MBF → Mcm1, and

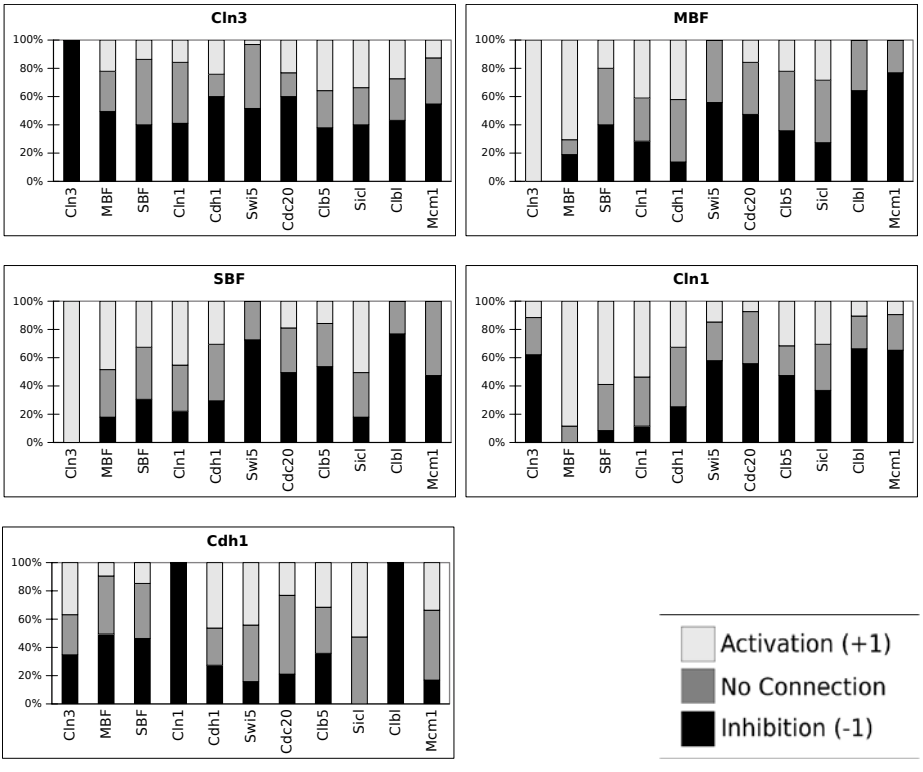


Fig. 5. Frequency of the relationships in the consistent networks. The statistics of inward connections to each gene from all other genes were created by the consecutively application of the three steps of the described algorithm and by a random determination of one connection. The determined connections exhibits only one color (black, white or gray), and the partially determined connections exhibit two colors. 100 networks were used for the statistical analysis. The results for the remaining genes are shown in Fig. 6.

others - transforming some connections from undetermined into partially determined connections. These results show that the cell cycle pathway constrains some connections, therefore restricting the whole network [12].

We can attribute this phenomenon to the high dependency that the determination of a network connection has on other connections. The three steps of the presented algorithm perform a search over the space of possibilities of the influence of a set of genes over a single gene. If one of these influences is *a priori* determined (or known), this result can bias other connections. For example, let us suppose that genes A and B have to produce a positive output over a gene C, according to some restriction imposed by the time-series data. If we already know that gene A has no relationship to gene C, gene B must have a positive relationship to gene C.

Therefore, this high dependency on the determination of a connection over the network makes the use of Fig. 5 and 6 very restricted. If we simply use a

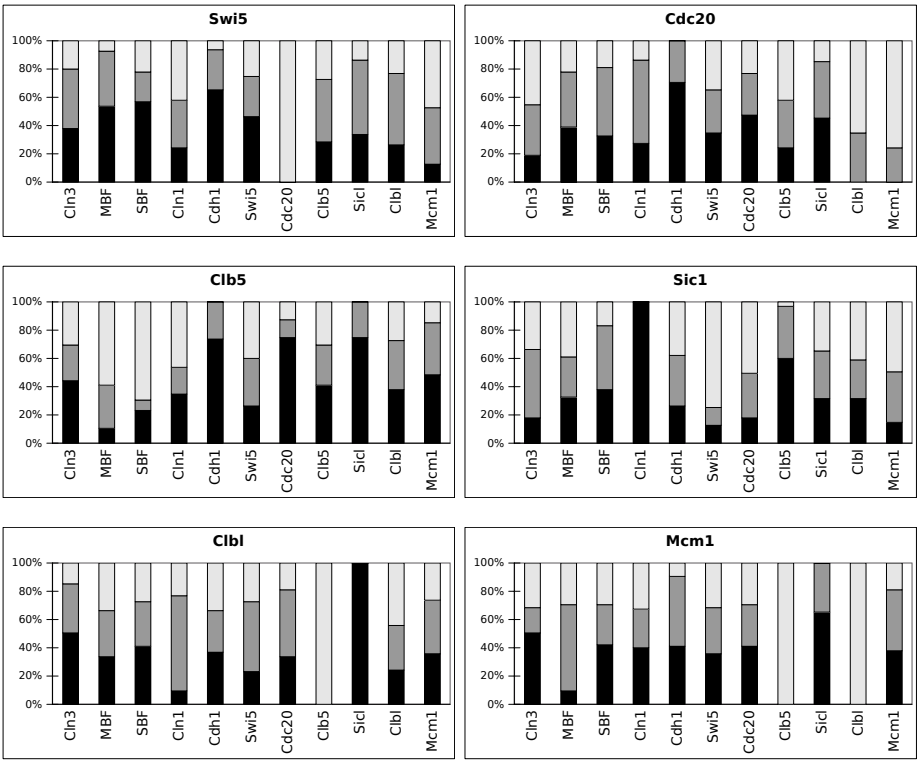


Fig. 6. Results for the genes Swi5, Cdc20, Clb5, Sic1, Clb1 and Mcm1

relationship with a high weight to be our "best guess" on the connection between two genes, this choice can constrain other relationships, leading the system to a more or less determined state, or even creating a network that is not consistent with the data.

We can say that Fig. 5 and 6 represent a good approximation of a "greedy" heuristic for finding one network. It can be done in the following way. Firstly, calculate the frequency of the connections of a set of consistent networks. Secondly, choose the most determined connection to be fixed with the relationship that has the greater weight. Thirdly, recalculate the set of networks and return to step one.

Another fact to be pointed out is the importance of the inferred partially determined connections. Although these connections can not be directly used to construct a network like the determined connections, it can guide some biological experiments, since a partially deterministic connection states that at least one type of relationship between two genes is not possible. We could use the frequencies generated in Fig. 5 and 6 to attribute a *strength of connection* to the relationship of a partially determined connection, e.g., the interference of Clb1 on SBF can be stated as 80% (or a probability of 0.8) of being an inhibition.

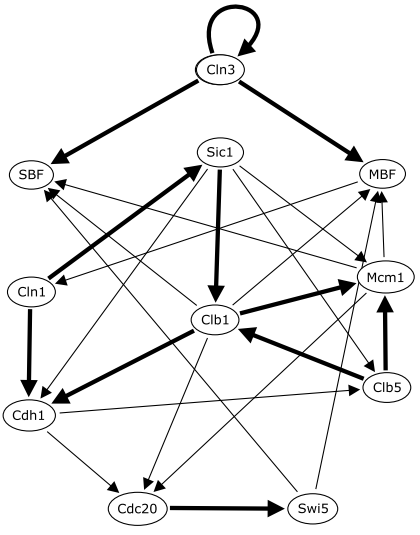


Fig. 7. The determined (bold arrows) and partially determined connections (light solid arrows) inferred by the consecutive application of the three steps of the algorithm

A closer look into the statistics raises also an interesting question: the network chosen by the nature would not be easily detectable? Or even better: would not the collected data be enough to constrain Fig. 5 and 6 into nature's choice? We could answer this question by pointing out a piece of information that makes a huge difference between our model and nature's choice: the chemical interactions between proteins. Evidently, some of the connections considered on many steps of the algorithm here presented can not exist due to chemical incompatibilities. In some sense, nature has more information to constrain its network than we do.

7 Conclusion and Future Research

This paper proposes an algorithm to perform analyses for discovering gene regulatory interactions from time-series data under the Boolean network model. In fact, the inference of gene regulatory networks is a one-to-many inverse problem in the sense that there may exist several networks consistent with the dataset. In order to analyze the gene interactions, we have generated several networks and considered only the consistent ones. We have applied our methodology to an artificial dataset that had been generated by a Boolean network that models the budding yeast cell cycle [14]. By this application, we have shown that this analysis of gene interactions could be a first step for gene relationships detection with a high flexibility to include biological knowledge.

A challenge always presented in any gene regulatory model is its usefulness. It would be very interesting if a model could help biological experiments in understanding gene interactions. The model here presented is capable of inferring

some of these connections from time-series data of gene expressions, and this inference process is helped by all *a priori* knowledge available.

Hence, an interesting feature to be added to our method would be the ability to indicate which connection should be verified in the wet lab to help determine others. As stated in the last section, the network connections are very dependent of each other, and the determination of one connection could constrain the whole network. What we envisage with our method is a model that points out which connections should be determined in the wet lab that in turn would constrain as many other connections as possible and consequently could facilitate some biological experiments. We are investigating the possibility to put our algorithm in the context of a *constraint solving problem* (CSP) [15]. There are CSP solver techniques that may help us to analyze the gene interactions as we did in this paper.

However, there are other characteristics to be sought that could constrain the network towards nature's choice. One feature not explored in this paper is the dynamical aspects of the network. There are indications, as stated by Kauffman [10], that nature would prefer networks with a small quantity of attractors - the gene pattern expression that leads the system to itself- and large basins of attraction - the set of gene pattern expressions that leads the system to one attractor. The network constructed by Li et al. [14] has these characteristics. Therefore, a connection statistics calculated only from networks with a few number of attractors - or other dynamical characteristic - could create a well established result.

Concluding, we think that the model here presented is a remarkable first step of the construction of a system to infer gene interactions. Our intention now is to test this procedure with another artificial data and, perhaps, biological data also; and to implement some topics presented in this section. We understand that any inference procedure can not have success if it does not contain biological and computational expertise, therefore the future steps of this research have to be centered on the difficulties of a wet lab, or its limitations.

References

1. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology 22, 437–467 (1969)
2. Shmulevich, I., Dougherty, E.R., Kim, S., Zhang, W.: Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. Bioinformatics 18(2), 261–274 (2002)
3. Zhang, S.Q., Ching, W.K., Ng, M.K., Akutsu, T.: Simulation study in Probabilistic Boolean Network models for genetic regulatory networks 1(3), 217–240 (2007)
4. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyze expression data. Journal of Computational Biology 7(3-4), 601–620 (2000)
5. Goodwin, B.C.: Temporal Organization in Cells; A Dynamic Theory of Cellular Control Process. Academic Press, London (1963)
6. Karlebach, G., Shamir, R.: Modelling and analysis of gene regulatory networks. Nature 9, 770–780 (2008)

7. Hartemink, A.J.: Reverse engineering gene regulatory networks. Nature 23(5), 554–555 (2005)
8. Hecker, M., Lambeck, S., Toepfer, S., van Someren, E., Guthke, R.: Gene regulatory network inference: Data integration in dynamic models - A review. BioSystems 96, 86–103 (2009)
9. Shmulevich, I., Dougherty, E.R., Zhang, W.: From Boolean to Probabilistic Boolean Networks as Models of Genetic Regulatory Networks. Proceedings of the IEEE 90, 1778–1792 (2002)
10. Kauffman, S.A.: The Origins of Order: Self-Organization and Selection in Evolution. Oxford University Press, Oxford (1993)
11. Huang, S.: Gene expression profiling, genetic networks, and cellular states: An integrating concept for tumorigenesis and drug discovery. J. Mol. Med. 77, 469–480 (1999)
12. Lau, K.Y., Ganguli, S., Tang, C.: Function Constrains Network Architecture and Dynamics: A Case Study on the Yeast Cell Cycle Boolean Network. Physics Review E 75(5), 1–9 (2007)
13. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B.: Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast *Saccharomices cerevisiae* by Microarray Hybridization. Molecular Biology of the Cell 9, 3273–3297 (1998)
14. Li, F., Long, T., Lu, T., Ouyang, Q., Tang, C.: The Yeast Cell-Cycle Network is Robustly Designed. PNAS of the USA 101(14), 4781–4786 (2004)
15. Tsang, E.: Foundations of Constraint Satisfaction. Academic Press, London (1993)

Residue Contexts: Non-sequential Protein Structure Alignment Using Structural and Biochemical Features

Jay W. Kim[1] and Rahul Singh[2,*]

[1] Department of Biology
[2] Department of Computer Science,
San Francisco State University, 1600 Holloway Ave., San Francisco, CA 94132, USA
rsingh@cs.sfsu.edu

Abstract. The study of non-sequential alignments, with different connectivity of the aligned fragments in the proteins being compared can offer a more complete picture of the structural, evolutionary and functional relationship between two proteins, than what is possible purely with sequential alignments. The design of techniques for non-sequential protein structure alignment therefore, constitutes an important direction of research. This paper introduces a novel method for non-sequential protein structure alignment involving three principle technical facets: (1) determination of the seed alignments not just by matching features from a single residue or considering well defined regions in the structure such as α–helices and β-strands, but through rich and robust descriptors that can capture the structural similarities of the local 3D environment around arbitrary residues of interest. (2) Scoring alignments using both geometric criterion (RMSD) as well as the biochemical characteristics of the residues. (3) An iterative chaining process which alternates between refinement and non-sequential extension stages to build a final alignment. The efficacy of the approach is demonstrated using the RIPC reference set which includes 40 structural pairs that are problematic to align. The performance of the method was found to be comparable or better than established techniques across the experiments.

1 Introduction

Given two structures, the problem of determining their structural similarity involves determining the correspondence of homologous residues between them such that each pair of aligned residues fulfils equivalent functional and structural roles. The ability to reason about structure, in a comparative setting, is important in providing a mechanistic understanding of the structure-property relationships that constitute the process of "life". Given that structure-level conservation is often much higher than sequence-level conservation, techniques for structure similarity can also provide clues to the unknown molecular function of a protein based on its structural similarity to one or more proteins of known function(s). Finally, structure similarity lies at the core of classifying protein structures and has been used in a variety of classification schemes such as SCOP, CATH and FSSP to name a few.

* Corresponding author. Research funded by NSF grant IIS-0644418.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 77–88, 2010.

The problem of structure matching or alignment has been widely studied during the past three decades leading to an increasingly deeper understanding of the challenges. For closely related proteins, different methods generally output consistent alignments. However recent studies have revealed significant inconsistencies between alignment methods for distantly related proteins [1]. Such inconsistencies arise when two related proteins display considerable structural variability resulting from the evolutionary accumulation of mutations [2]. Determining non-sequential alignments, with different connectivity of the aligned fragments in the proteins being compared, constitute an intriguing problem in this context. One such example is a circularly permutated protein where the evolutionary divergence from an ancestor has resulted in a change in domain ordering. In such cases, an accurate alignment of a circularly permutated regions, region swaps and β-hairpin flips requires that a matching/alignment technique align individual residues or fragments while disregarding their natural sequence and order. A commonly encountered example is that of the Rossmann structure motif, which comprises of four α–helices and four β–strands and can be found with different SSE connectivity. It should be noted that proteins requiring non-sequential alignments comprise a non-trivial proportion of known protein structures (estimated to be between 17.4% and 35.2% of all alignments [3]). In proposing a solution to the non-sequential alignment problem, the method proposed in this paper seeks to focus on the following two important sub-problems:

- Design of algorithms for determining the initial (seed) alignments, based on which, the ultimate alignment is obtained. The goal is to determine the seed alignments, not just by matching features from a single residue or considering well-defined regions in the structure such as α–helices and β-strands, but through rich and robust descriptors that can capture the structural similarities of the local 3D environment around arbitrary residues of interest.
- Determination of the alignments, not just based on geometric criteria (such as RMSD), but also by involving biochemical characteristics of the residues.

To motivate the importance of the first sub-problem, a brief review of different non-sequential alignment techniques is necessary. These methods can be broadly classified into two groups based on how the initial (seed) correspondences between substructures of the two proteins are detected: residue-based seed matching (RSM) methods and secondary structure element-based (SSE) methods. Examples of RSM methods include STSA [4], and our method. In such methods, the initial correspondences are obtained by modelling and matching substructures in terms of their geometric properties (though in our method, we employ both geometric and biochemical characteristics). In SCALI [5], correspondence is established when the fragments being matched contain greater than five residues, do not have any gaps/insertions, do not have residues with backbone angles differing by greater than $90^{o,}$ and are not part of longer fragments considered earlier. In contrast to RSM methods, SSE-based methods ameliorate the complexity of finding initial correspondences by focussing on similar secondary structure elements (α-helices and β-strands). In GANGSTA [6], pair contacts and relative orientations between SSE are maximized using a genetic algorithm. Next, residue pair contacts between the best SSE-alignment are optimized. In SSM [7], correspondences are obtained by graph matching based on SSEs. In addition to sequential alignment, the method allows

complete non-sequential alignment, where the connectivity is neglected, and a "soft" alignment, where the general order of SSEs is retained with the provision that any number of intervening unmatched/missing SSEs are allowed. Finally, TOPOFIT [3] constitutes a technique which does not clearly fall in either of the aforementioned two groups. In TOPOFIT, Delaunay triangulation of the points representing the proteins is used to construct tetrahedrons which are subsequently matched in terms of shape, volume, and backbone topology to find the seed correspondences. In summary, methods that use SSEs to find seed correspondences, ultimately treat the protein structure at a coarser level of granularity, than what is possible at the residue or atomic level. While this allows ameliorating the match complexity, it is possible to miss seed correspondences that do not fall in regions corresponding to well-defined SSEs. In contrast, such a risk is inherently lower in RSM methods which treat the structure at a finer granularity. However, this does require solving a more complex correspondence problem.

2 Proposed Method

We approach the problem of non-sequential structure alignment, in context of the two aforementioned sub-problems, as a three-step process:

(1) *Determining the initial correspondence (seed determination)*: The initial correspondence provides a (possibly coarse) match between similar substructures in the two molecules and can be thought of as the first approximation of the alignment. To determine the initial match we propose a novel rotation invariant and geometrically rich local structure descriptor, which we call the *residue context*. The residue context is a quantized description of the *distribution* of atoms or residues in 3D space with respect to a given point on the protein structure. In this work, the residue context is determined at each C_α atom on the protein backbone. Thus, solving the initial correspondence problem reduces to finding for each C_α atom on one structure, the corresponding C_α atom on the other structure that has the most similar residue context. We formulate the problem of determining the similarity of two residue contexts in terms of the transportation problem, which is a special case of linear programming. This allows us to use the Earth Mover's Distance (EMD) [12] to efficiently address this question. A fundamental advantage of our matching formulation is that it naturally overcomes representation variations that occur due to quantization.

(2) *AFP Generation using Structural and Bio-Chemical information*: In this step, regions are identified using a geometric-fit criteria and analyzed based on the biochemical agreement of the aligned residues, to obtain aligned fragment pairs (AFP). Inspection through this "double lens" of geometric and physicochemical properties raises the likelihood of only procuring desirable AFPs, which serve as interchangeable building blocks for the construction of the final alignment.

(3) *AFP Chaining and Refinement*: In the final step, the AFPs undergo stages of assembly and restructuring as determined by a composite alignment score, to obtain longer and more accurate alignments. The chaining and refinement stages are iterative such that hard correspondences are not assigned until the alignment score does not improve further. The final alignment can be non-sequential or sequential and is driven solely by the structures being compared.

2.1 Definition of Residue Context

In the first step of the proposed method, similar substructures in the two molecules are determined by capturing the structural similarities of the local 3D environment around arbitrary residues of interest through their residue contexts. The design of this descriptor is motivated by research in computer vision on shape recognition [9]. The underlying insight utilizes results from stereopsis indicating that determining correspondences between shapes is easier with rich local descriptors (such as the one proposed in [9] as well as residue context) as opposed to features that are dependent on single shape primitives. Our research extends to 3D molecular structures, the basic idea of shape-context descriptors introduced in [9], namely that given a point on a shape, the distribution of other shape points around this point constitutes a compact, yet highly discriminative descriptor of the local shape geometry.

The notion of residue context, as proposed by us can be described as follows: given the protein backbone defined through the 3D coordinates of its constituent C_α-atoms, and a reference C_α-atom, consider the set of n-1 vectors originating from the reference C_α atom to all the other C_α-atoms of the backbone. These vectors describe the configuration of the entire backbone shape relative to the reference atom and can be thought of as to constitute its local shape context in 3D space. It may be noted that the set of n-1 vectors constitutes a rich description, since, as n (the number of C_α-atoms) increases, the representation of the backbone shape becomes exact. The distribution of the vectors centered at a reference C_α-atom can be succinctly represented using a 3D spherical histogram centered at the reference atom. Further, each of the vectors can be defined by three parameters in a spherical coordinate system: the radial distance $\vec{r}$ corresponding to the distance between the reference C_α-atom and another C_α-atom, the azimuthal (longitude) angle θ in the x-y plane from the x-axis with $0 \le \theta \le 2\pi$ and the polar (latitude) angle ϕ from the z-axis with $0 \le \phi \le \pi$. Following [9], we require the 3D spherical histogram centered on the reference C_α-atom to have the following two properties:

- The descriptor needs to be more sensitive to nearby residues than residues that are farther away. This property corresponds to the importance of proximity in defining intermolecular interactions. To ensure this property, the magnitude of r is logarithmically discretized and the longitude angle is uniformly discretized in the range [0, 2π].
- Bins equidistant from the center should cover the same surface area. This property ensures that the representation is isotropic in space. To support it, the latitude angle $\phi \in [-\pi/2, \pi/2]$, is discretized non-uniformly, such that each ϕ_i satisfies the relationship in Eq. (1), where the righthand side denotes the i^{th} fraction of the surface area of the upper hemisphere:

$$\int_{\theta=0}^{2\pi} \int_{\phi=0}^{\phi_i} r^2 \cos\phi \, d\phi \, d\theta = \frac{i}{N} 2\pi r \tag{1}$$

From Eq. (1), the required discretization of the latitude angle is $\phi_i = \arcsin(i/N)$.

Given a reference C_α-atom and a spherical histogram centered on it, the residue context of this C_α-atom is constituted by the distribution, within the bins of the

histogram, of the other n-1 C_α-atoms of the protein backbone, that fall within the radius r. Specifically, for a reference atom $C\alpha_j$, the histogram H_j of the relative coordinates of the remaining n-1 atoms is given as:

$$H_j(k) = |\{C\alpha_i \neq C\alpha_j : (C\alpha_i - C\alpha_j) \in bin(k)\}| \quad (2)$$

In Eq.(2), $C\alpha_j$ is the reference atom, $C\alpha_i$ indexes the set of neighboring atoms located within the radius r of the reference atom, and |.| denotes the number of the neighboring reference C_α-atoms that fall within the k^{th}-bin of the histogram. An example, illustrating this concept is shown in Figure 1. Finally, given a molecule M, consisting of m alpha-Carbon atoms $C\alpha_i$, $i=1,\ldots,m$, its residue context-based description, denoted as $R(M)$, consists of the set of m histograms H_i, with each centered on one of the alpha-Carbon atoms of M: $R(M) = \{H_1, H_2, \ldots, H_m\}$.

One important practical issue in defining residue contexts is that of the context scale (size), which is specified by the choice of the context radius r. A large r, which considers a more global environment around each residue, can be useful for simple alignments consisting of two proteins with high sequence and structural similarities. Conversely, a smaller r may be necessary for making difficult non-sequential alignments and aligning two proteins of low sequence and structural similarities. In such cases, a large residue context may be counter-productive since it would incorporate variances due to extensive insertions, deletions, repetitions, and conformational variability. In subsection 2.4 we further address the issue of automatic scale selection for alignment.

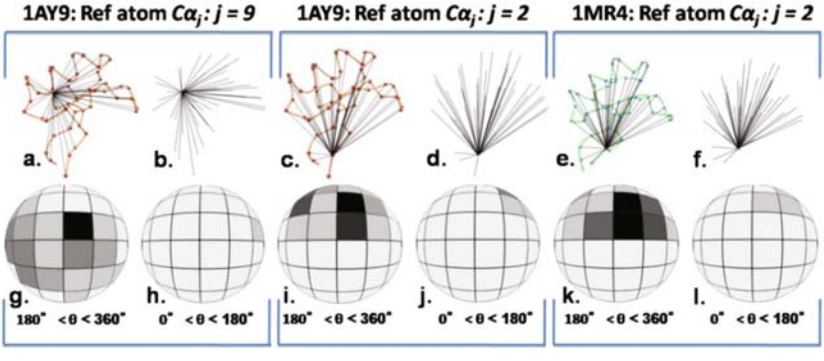


Fig. 1. 3D backbone representations of 1AYJ **(a,c)** and its homolog 1MR4 **(e)** where positions of C_α-atoms are shown as red or blue dots. 3D vectors originating from reference residue j=9 and j=2 of 1AYJ to all other C_α-atoms are shown in **(b)** and **(d)**,respectively. The corresponding "front" (180° <θ< 360°) and "back" (0° <θ< 180°) views of the residue contexts at these positions are shown in figures **(g-h)**, and figures **(i-j)**. One may note that the residue contexts are clearly distinct for these positions. In **(f)** the 3D vectors originating from reference residue j=2 of 1MR4 are shown. Since 1MR4 is a homolog of 1AYJ, we expect residue contexts at similarly located C_α-atoms in 1AYJ and 1MR4 to be similar. The "front" and "back" views of the residue contexts at j=2 of 1MR4 are shown in figures (k) and (l). The reader may note the similarity of the residue contexts (at reference residue j=2) for 1AYJ and 1MR4 and be comparing the "front" and "back" views from figures **(i)** and **(j)** with the corresponding views in figures **(k)** and **(l)**. In all the figures, darker bins are more heavily populated.

2.3 Efficient Matching of Residue Contexts

The problem of comparing residue contexts can directly be interpreted as that of matching two histograms. Several measures have been proposed to address this problem and they can be broadly classified into two categories. Most fall into the first category of bin-by-bin dissimilarity measures. This includes χ^2 statistics (used in [9] to compare 2D shape contexts), histogram intersection, L_p distances, Kullback-Leibler divergence, Jeffry divergence, and Jensen-Shannon divergence. A fundamental assumption underlying these techniques is that the domain of the histograms can be aligned. However, in practice, this assumption can be violated due to noise, sub-optimal quantization (binning), different number of bins, or the inherent nature of the data. The second category of measures is called cross-bin measures. Cross-bin measures utilize the ground distance between representative features in different bins to compare both aligned and non-aligned bins. The earth-mover's distance (EMD) [8] is an example of such a measure and is used by us.

Given two residue contexts, defined in terms of their respective histograms P and Q, one of them can be interpreted as a mass distribution spread on the underlying space and the other as a collection of holes in that same space. If a unit of work corresponds to transporting a unit of mass by a unit of ground distance, then the matching problem can be defined as determining the least amount of work required to fill the holes. This precisely corresponds to the EMD between the two distributions. Following [12], we formalize our problem as follows: Let the first histogram be represented by a set of tuples $P = \{\langle \boldsymbol{p}_1, w_{p1}\rangle, \langle \boldsymbol{p}_2, w_{p2}\rangle, \ldots, \langle \boldsymbol{p}_m, w_{pm}\rangle\}$, where the i^{th} bin is represented by the tuple $\langle \boldsymbol{p}_i, w_i\rangle$ with $\boldsymbol{p}_i$ denoting an appropriately chosen bin representative (such as its mean, centroid, or medoid) and w_{pi} the weight of the i^{th} bin, given by the fraction of residues from the context that fall into this bin. Similarly, let $Q = \{\langle \boldsymbol{q}_1, w_{q1}\rangle, \langle \boldsymbol{q}_2, w_{q2}\rangle, \ldots, \langle \boldsymbol{q}_m, w_{qm}\rangle\}$ be the tuple set representing the second histogram and d_{ij} denote the ground distance between bins $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$ (we use the Euclidean distance as the ground distance). Matching the residue contexts by computing the EMD requires solving the following minimization problem, where f_{ij} denotes the flow between $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$:

$$\arg\min_{f_{ij}} \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} f_{ij} \tag{3}$$

The minimization is subject to the constraints (4) – (7) below, where the constraint (4) ensures that the mass is moved in only one direction, constraint (5) and (6) ensure that the mass sent by bins in P and the mass received by bins in Q is limited to their weights, and constraint (7) requires that the maximum possible amount of mass is moved.

$$f_{ij} \geq 0, i = 1,2,...,m;\ j = 1,2,...,n \tag{4}$$

$$\sum_{j=1}^{n} f_{ij} \leq w_{p_i}, i = 1,2,...,m \tag{5}$$

$$\sum_{i=1}^{m} f_{ij} \leq w_{q_j}, j = 1,2,...,n \tag{6}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{m} w_{p_i}, \sum_{j=1}^{n} w_{q_j}\right) \tag{7}$$

Given the optimal flows f_{ij} obtained from solving the transportation problem as described above, the EMD between the two residue contexts is defined as:

$$EMD(P,Q) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} f_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}} \tag{8}$$

In Eq. (8), the numerator denotes the resulting work and the denominator describes the total flow.

2.4 Determining the Scale of the Residue Context

The size of the environment around each residue of a protein chain is determined by the radius r which is logarithmically discretized. For arbitrary alignments it is not possible to determine, *a priori*, the value of r for the optimal context size. We use a data driven procedure where the optimal value of the radius is defined as the one which best captures the similarity between two sub-chains across all possible values of the radius. This is done by computing n cost matrices C_n, which correspondingly store the costs associated with residue matches using contexts of varying radii r_n. The optimal cost matrix corresponds to the most similar contexts given by the lowest matching costs (Eq. 9) across the radii.

$$C_{ij}^{opt} = \min\left(\frac{C_1}{a_1+b_1}, \frac{C_2}{a_2+b_2}, ..., \frac{C_i}{a_i+b_i}\right) \tag{9}$$

$$M_{ij} = low + \frac{-C_{ij}^{opt} + \min}{\max - \min}(high - low) \tag{10}$$

In Eq. (9), a_n and $b_n$are the respective number of residues from protein chain A and B with context radii r_n. Next, the entries in the optimal cost matrix are normalized to lie in the interval [*low*, *high*], with *low* set to -100 and high set to 100 using Eq. (10). Matching two histograms whose bins are identically populated yields a score of 100. An example illustrating the intuition underlying the notion of residue context-based description and matching is described in Fig. 1.

2.5 AFP Generation and Scoring

Given two structures to be aligned as input, we define an aligned fragment pair (AFP) as a correspondence of residues between fragments from each structure. Our definition of an AFP differs from the original definition given by Shindyalov and Bourne [10] in that we allow single-residue gaps in either fragments' sequence to accommodate for single-residue insertions/deletions encountered in aligning structures displaying low sequence similarity. Larger gaps are naturally accommodated by the mechanics of our AFP chaining algorithm described later. Given the entire set of residues $\{p\}$ from molecule A and $\{q\}$ from molecule B, for the identification of AFPs, we first locate all triplets of residue pairs $\tau_{ij} = \{(p_{i-1},q_{j-1}),(p_i,q_j),(p_{i+1},q_{j+1})\}$ which occur continuously along a diagonal of the similarity matrix M such that $M_{i-1,j-1}$, $M_{i,j}$, and $M_{i+1,j+1}$ all exceed an AFP initiation threshold value t. The set of residues $\{q\}$ is transformed such that the three pairs of residues defined by τ are optimally superimposed and thedistances between the residues of $\{p\}$ and $\{q\}$ are stored in a matrix D_{ij}. Next, each triplet is extended in both the N-terminal and C-terminal directions based on the following two conditions, as long as the EMD score $M_{i+1,j+1}$ stays below an extension threshold $e$and the aligned distance of the extended correspondence does not exceed the aligned

distance of the C-terminal correspondence (prior to extension) by 3Å. The two conditions are: ***cond1:*** $M_{i+2,j+1} \geq M_{i+1,j+1}$ && $M_{i+3,j+2} \geq M_{i+2,j+2}$ && $D_{i+2,j+1} < D_{i+1,j+1}$; ***cond2***: $M_{i+1,j+2} \geq M_{i+1,j+1}$ && $M_{i+2,j+3} \geq M_{i+2,j+2}$ && $D_{i+1,j+2} < D_{i+1,j+1}$. If only ***cond1*** holds, the correspondence (p_{i+2},q_{j+1}) is added to the AFP. Similarly, if only ***cond2*** holds, (p_{i+1},q_{j+2}) is added. Finally, if both conditions hold, then the correspondence with the lowest RMSD is added. The resulting set of AFPs $F = \{f\}$ is filtered to ensure that an AFP contains a minimum of 4 residue correspondences. Further, AFPs that are completely contained within a larger AFP are discarded. Note that although each $f \in F$ is unique in its entirety, AFPs are allowed to extend freely with partial overlap to avoid introducing bias based on the initial triplet locations. At this point, any two or more overlapping AFPs represent a collection of residue pair correspondences whose final alignment path has not yet been determined. This uncertainty is resolved during the subsequent AFP chaining step in section 2.6.

The AFPs are next ranked using an AFP alignment score *AS* (Eq. 11) which is the weighted sum of two component scores; the structural score *SS* is defined simply as the sum of the similarity scores along the length *l* of the AFP (Eq. (12)).

$$AS = w_{SS} + w_{BS} \qquad (11) \qquad\qquad SS = \sum_{i,j=1}^{l} M_{ij} \qquad (12)$$

The second component of the alignment score is the biochemical score *BS*. The biochemical score captures the likelihood of the evolutionary occurrence of each pair-wise amino acid substitution as suggested by the residue correspondences defined by the AFP. Its use is motivated by the assumption that among structurally and functionally conserved proteins, the frequency of amino acid substitutions at a given site is correlated with the physicochemical similarities between exchanged amino acids. Thus by rewarding biochemical agreement between aligned residues, we seek to select a pool of AFPs that contain conserved functional alignments between two proteins. To compute the biochemical score *BS*, the Blosum62 matrix is used to estimate the likelihood of occurrence of each possible pair-wise amino acid substitution. Depending on the likelihood of substitution, each pair of aligned residues in an AFP earns a predefined numerical score towards the total biochemical score for the AFP as follows: the Blosum62 matrix values are first normalized over the interval [*low*, *high*] with *low* set to -100 and *high* set to 100. The normalized Blosum62 matrix B_{ij} is computed using the following equation and values less than zero are reset to zero:

$$B_{ij} = low + \frac{-Blosum_{ij}^{opt} + \min}{\max - \min}(high - low) \qquad (13)$$

In Eq. (13), *max* and *min* denote the highest and lowest values found in the Blosum62 matrix. In our current investigations, the SS and BS components are given equal weight, that is, $w_{SS} = w_{BS} = 1$. However, these weights can be changed, if needed, to emphasize either of the components.

2.6 Chaining and Refinement Using Structural and Biochemical Scores

Given two molecules *A* and *B*, the chaining process starts with the seed alignments captured by the AFPs. A crucial challenge in extending the seed alignments, is that of

avoiding spurious alignments. Specifically, during protein alignment numerous short AFPs of length 4~6 residues can be encountered which can be superimposed at low RMSD values. However, such alignments are often misleading. For example, two β-strands of length 4~6 or two segments consisting of 1~2 turns of an α-helix are often structurally similar in any two protein. Thus, a strategy that simply minimizes RMSD can lead to incorrect alignments. We therefore chain the AFPs by using the alignment score *AS* defined earlier. It may be noted that this score consists of the EMD score (capturing topological local shape similarity) and the biochemical scores (reflecting biochemical similarity). The initial chain is constructed as follows: we begin with the set of k AFPs denoted as $\{f^k\}$. This set is sorted by the *AS* score and the highest scoring AFP is denoted as f^1. The initial alignment is $c^{init}=f^1$. Next, an optimal (in the least square sense) Euclidean transformation T is calculated to align the subset of residues $\{q\}$ from molecule B with the corresponding residues $\{p\}$ from molecule A, where the correspondences are given by c^{init}. This optimal transformation T is applied to molecule B to give B^* and the RMSD between subset $\{p\}$ and transformed subset $\{q^*\}$ from B^* is stored as the chain RMSD. Further, a chain alignment score *CS* is determined as follows:

$$CS = \frac{w_{SS} + w_{BS}}{RMSD} \tag{14}$$

Subsequently, the remaining APFs in $\{f^k\}$ are treated as follows. Any AFPs that overlap with the residues contained in the current chain are discarded. Thus each residue p_i contained in the chain must have a unique corresponding residue q_j and vice versa. A non-overlapping AFP is added to the chain only if its addition increases the alignment score *CS*. After all AFPs have been considered, the resulting chain is stored in the set of chains *C*. The initial chaining process is repeated, substituting the next highest scoring AFP denoted as f^2 for the initial seed alignment, and iterated for each of the top 50 AFPs. The highest scoring chain from *C* is passed to the refinement step.

For the refinement of a chain, we first reduce $\{f^k\}$ to only include AFPs which overlap the immediate vicinity of a residue correspondence stored in the chain. Given a correspondence (p_i,q_j), its immediate vicinity is defined on an i x j alignment matrix as the area enclosed by: $(p_{i-\delta},q_{j-\delta})$,$(p_{i-\delta},q_{j+\delta})$,$(p_{i+\delta},q_{j-\delta})$, and $(p_{i+\delta},q_{j+\delta})$. In all our experiments, δ is set to 3. As in the initial chaining step, each AFP in the reduced set is considered and included only if *CS* increases after its addition while giving priority to the new correspondences in case of overlap. Thus any redundant residue and its corresponding partner are removed from the current chain before the new correspondence given by the AFP is added. The chaining and refinement steps are iterated using the refined chain as input for each chaining step. The process is stopped when the chain alignment score *CS* converges or changes between successive iterations become smaller than a predefined threshold. Before the final alignment is output, the N and C-terminal correspondences of chained AFPs are briefly extended as described in section 2.5.

3 Experimental Investigations and Results

The RIPC set comprises 40 structural pairs that are problematic to align [1]. Each pair in this set is characterized by repetitions, extensive insertions and deletions, circular

permutations, and/or conformational variability. Human-curated reference alignments based on conservation of sequence and function are provided for 23 out of 40 protein pairs. Agreement to the reference alignments is measured for each pair by the fraction of correctly aligned residues, f_{CAR}, numerically defined as:

$$f_{CAR} = \#\text{ of correctly aligned residues} / \#\text{ of reference pairs} . \quad (15)$$

We compared our performance against 3 non-sequential alignment methods (GANGSTA, TOPOFIT, STSA), and 4 sequential alignment methods (DALI [11], CE [8], MATT [12], FATCAT [13]) (Fig. 2). MATT and FATCAT are also flexible aligners that allow twists and translations to the protein backbone to accommodate for conformational variability.

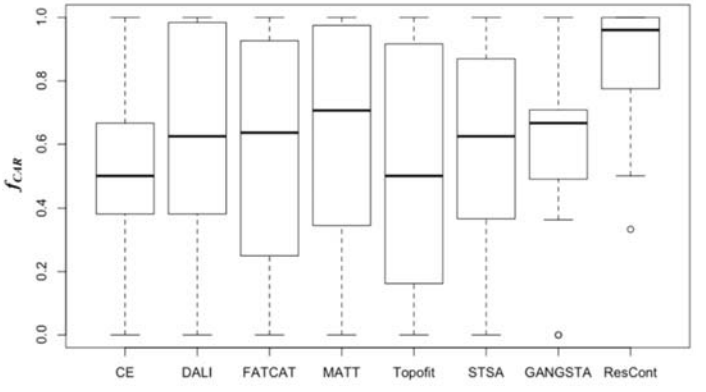


Fig. 2. Comparison of various methods' performances on the RIPC reference set. Box and whisker plot properties are as follows: bottom whisker – min sample, lower box boundary – 1st quartile, bolded line – median, upper box boundary – 3rd quartile, top whisker – max sample. The statistical median for each method was calculated from 23 samples (alignments) where the f_{CAR} for each alignment served as a measure of agreement with the RIPC reference.

Among the methods tested, Residue Context showed the highest agreement with the reference set (median = 96%) and Matt was second highest (median = 71%). Residue Context was the only method which correctly aligned at least one reference pair for each of the 23 alignments. The lowest f_{CAR} obtained using our method was for the alignment of an E6AP-UbcH7 complex (d1d5fa_) to a HECT domain E3 ligase (d1nd7a_) for which 4 of 6 reference pairs were missed. The alignment requires accounting for considerable conformational variability to correctly align all reference pairs. DALI and FATCAT managed to correctly align all 6 reference pairs. The alignment of an L-2-haloacid dehalogenase (d1qq5a_) and *E. Coli* CheY (d3chy__) confounded most methods. This alignment involves a circular permutation, and also extensive insertions are present in d1qq5a_ with respect to d3chy__. We found that non-sequential methods as well as sequential methods produced inconsistent alignments. Only Residue Context was able to align all 3 reference residues correctly. On the other hand, Topofit and STSA missed all 3 reference residues (Table 1). When only considering non-sequential alignments, Residue Context had the highest median and mean among 4 methods at 94% and 89%, respectively. Topofit had the 2nd highest

median, but also displayed the greatest inconsistency between alignments as evidenced by the disparity between its Q3 (94%) and Q1 (25%) values. Residue Context was the most consistent (Q3 = 100%; Q1 = 81%) method across the dataset.

Table 1. Statistical comparison of performances of three non-sequential methods on non-sequentially related pairs from the RIPC set.The mean, median, 1st quartile (Q1), and 3rd quartile (Q3) were calculated using f_{CAR} values from 10 non-sequential alignments of the RIPC set. The highest scores (including ties) for each alignment and statistical category are bolded.

	Residue Context					**STSA**				
Aligned Pair	**Length**	**RMSD**	**Aligned**	**Total**	f_{CAR}	**Length**	**RMSD**	**Aligned**	**Total**	f_{CAR}
d1nkl__-d1qdma1	73	2.61	54	72	0.75	74	2.26	72	72	**1.00**
d1nls__-d2bqpa_	221	1.44	6	6	**1.00**	212	1.50	2	6	0.33
d1qasa2-d1rsy__	113	1.87	72	75	**0.96**	111	1.94	67	75	0.89
d1b5ta_-d1k87a2	227	3.80	5	8	**0.63**	188	2.84	5	8	**0.63**
d1jwyb_-d1puja_	148	3.34	11	12	**0.92**	116	2.34	9	12	0.75
d1jwyb_-d1u0la2	124	3.69	11	11	**1.00**	99	2.68	8	11	0.73
d1nw5a_-d2adma_	166	3.96	13	13	**1.00**	120	2.57	11	13	0.85
d1gsa_1-d2hgsa1	83	3.29	4	5	**0.80**	229	2.59	2	5	0.40
d1qq5a_-d3chy__	107	3.48	3	3	**1.00**	92	2.73	0	3	0.00
d1kiaa_-d1nw5a_	162	3.83	10	12	0.83	90	3.37	0	12	0.00
	Mean	**Q1**	**median**	**Q3**		**mean**	**Q1**	**median**	**Q3**	
	0.89	**0.81**	**0.94**	**1.00**		0.56	0.35	0.68	0.82	
	GANGSTA					**Topofit**				
Aligned Pair	**Length**	**RMSD**	**Aligned**	**Total**	f_{CAR}	**Length**	**RMSD**	**Aligned**	**Total**	f_{CAR}
d1nkl__-d1qdma1	74	2.41	72	72	**1.00**	56	1.65	28	72	0.39
d1nls__-d2bqpa_	222	3.23	4	6	0.67	212	1.01	6	6	**1.00**
d1qasa2-d1rsy__	115	2.95	44	75	0.59	105	1.16	71	75	0.95
d1b5ta_-d1k87a2	181	3.34	5	8	**0.63**	134	1.85	1	8	0.13
d1jwyb_-d1puja_	137	2.83	9	12	1.75	108	1.65	11	12	**0.92**
d1jwyb_-d1u0la2	111	2.60	11	11	**1.00**	99	1.58	11	11	**1.00**
d1nw5a_-d2adma_	146	2.98	13	13	**1.00**	93	1.61	11	13	0.85
d1gsa_1-d2hgsa1	75	2.38	2	5	0.40	66	1.44	1	5	0.20
d1qq5a_-d3chy__	101	3.36	2	3	0.67	63	1.65	0	3	0.00
d1kiaa_-d1nw5a_	150	2.94	8	12	0.67	132	1.73	11	12	**0.92**
	Mean	**Q1**	**median**	**Q3**		**mean**	**Q1**	**median**	**Q3**	
	0.74	0.64	0.67	0.94		0.64	0.25	0.89	0.94	

In the next experiment we compared the performances of three non-sequential methods on the alignment of structures involving the Rossmann fold (data from [3]). GANGSTA generally produced the longest alignments while Residue Context produced alignments of comparable lengths at significantly lower RMSDs (Table 2). Topofit generated significantly shorter alignments at lower RMSDs. Examples of 3D structural representations of alignments generated by Residue Context are shown in Figure 3.

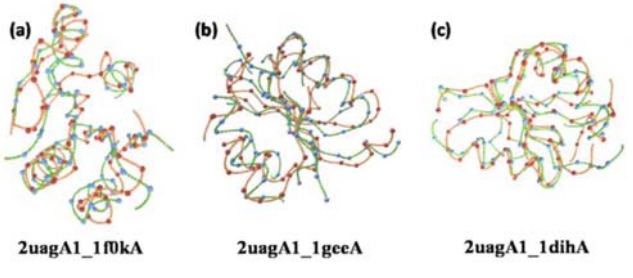


Fig. 3. 3D structural representations of 3 non-sequential alignments involving the Rossmann fold. (a-c) obtained using the Residue Context method

Table 2. Comparison of three methods on alignments involving the Rossmann fold

	Residue Context			GANGSTA			Topofit		
Structures	**Length**	**RMSD**	**Length/RMSD**	**Length**	**RMSD**	**Length/RMSD**	**Length**	**RMSD**	**Length/RMSD**
2uagA1_1f0kA	83	2.63	31.6	85	3.52	24.1	41	1.34	30.6
2uagA1_1geeA	85	2.93	29.0	89	3.28	27.1	60	1.56	38.5
2uagA1_1dih_1	83	2.46	33.7	82	3.07	26.7	63	1.60	39.4

4 Conclusions

This paper considers the problem of non-sequential protein structure alignment. We have presented a novel approach that involves determining initial (seed) correspondences using a rich descriptor that can capture structural similarities of the local 3D environment around arbitrary residues of interest. Based on these seed correspondences the alignments are constructed using geometric and biochemical characteristics of the involved residues. Experiments indicate that, in terms of alignment quality, the proposed method either exceeds or is comparable with leading methods at the state of the art.

References

1. Mayr, G., Domingues, F.S., Lackner, P.: Comparative Analysis of Protein Structure Alignments. BMC Structural Biology 7(50), 564–577 (2007)
2. Grishin, N.: Fold change in evolution of protein structures. J. Struct. Biol. 134, 167–185 (2001)
3. Ilyin, V.A., Abyzov, A., Leslin, C.M.: Structural alignment of proteins by a novel TOPOFIT method, as a superimposition of common volumes at atopomax point. Protein Sci. 13(7), 1865–1874 (2004)
4. Salem, S., Zaki, M.J., Bystroff, C.: Iterative Non-Sequential Protein Structural Alignment. Journal of Bioinformatics and Computational Biology 7(3), 571–596 (2009)
5. Yuan, X., Bystroff, C.: Non-sequential structure-based alignments reveal topology-independent core packing arrangements in proteins. Bioinformatics 21(7), 1010–1019 (2005)
6. Kolbeck, B., May, P., Schmidt-Goenner, T., Steinke, T., Knapp, E.W.: Connectivity independent protein-structure alignment: a hierarchical approach. BMC Bioinformatics 7, 510 (2006)
7. Krissinel, E., Henrick, K.: Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. ActaCrystallogr D BiolCrystallogr 60(1), 2256–2268 (2004)
8. Rubner, Y., Tomasi, C., Guibas, L.J.: A Metric for Distributions with Applications to Image Databases. In: Proceedings of the 1998 IEEE Conf. on Computer Vision, pp. 59–66 (1998)
9. Belongie, S., Malik, J., Puzicha, J.: Shape Matching and Object Recognition Using Shape Contexts. IEEE Trans.on Pattern Analysis and Machine Intelligence 24, 509–522 (2002)
10. Shindyalov, I.N., Bourne, P.E.: Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. Protein Engineering 11(9), 739–747 (1998)
11. Holm, L., Sander, C.: Protein structure comparison by alignment of distance matrices. J. Mol. Biol. 233(1), 123–138 (1993)
12. Menke, M., Berger, B., Cowen, L.: Matt: local flexibility aids protein multiple structure alignment. PLoSComput Biol. 4(1), e10 (2008)
13. Ye, Y., Godzik, A.: Flexible structure alignment by chaining aligned fragment pairs allowing twists. Bioinformatics 19(2), 246–255 (2003)

Essential Proteins Discovery from Weighted Protein Interaction Networks*

Min Li[1], Jianxin Wang[1,2], Huan Wang[1], and Yi Pan[1,2]

[1] School of Information Science and Engineering,
Central South University, Changsha 410083, P.R. China
[2] Department of Computer Science,
Georgia State University, Atlanta, GA 30302-4110, USA

Abstract. Identifying essential proteins is important for understanding the minimal requirements for cellular survival and development. Fast growth in the amount of available protein-protein interactions has produced unprecedented opportunities for detecting protein essentiality on network level. A series of centrality measures have been proposed to discover essential proteins based on network topology. However, most of them treat all interactions equally and are sensitive to false positives. In this paper, six standard centrality measures are redefined to be used in weighted network. A new method for weighing protein-protein interactions is proposed based on the combination of logistic regression-based model and function similarity. The experimental results on yeast network show that the weighting method can improve the performance of centrality measures considerably. More essential proteins are discovered by the weighted centrality measures than by the original centrality measures used in unweighted network. Even about 20% improvements are obtained from closeness centrality and subgraph centrality.

Keywords: essential protein, protein interaction network, centrality.

1 Introduction

In the post-genome era, the developments of high-throughput methods, such as yeast-two-hybrid and mass spectrometry, have produced vast amounts of protein-protein interaction data, which make it possible for us to study genes and proteins in network level [1]. The corresponding protein interaction networks provide useful insights into cellular effects and functional associations between proteins[2]. Recently, much attention has been paid to the study of the properties of protein interaction networks, including the global properties

* This work is supported in part by the National Natural Science Foundation of China under Grant No.60773111, the Ph.D. Programs Foundation of Ministry of Education of China No. 20090162120073, the U.S. National Science Foundation under Grants CCF-0514750, CCF-0646102, and CNS-0831634, and the Program for Changjiang Scholars and Innovative Research Team in University No. IRT0661.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 89–100, 2010.

such as"scale-freeness" [3] and "small-world behavior" [4], modularity [5,6] and disassortativity [7].

The analysis of protein interaction networks is to discover the interrelationship between the topological properties and the biological characteristics. An intriguing question is whether the essentiality of a protein can be explained by its placement in the network. A protein is said to be essential for an organism if a knock-out results in lethality or infertility, i.e., the organism cannot survive without it [8,9]. It has been observed in several species, such as *Saccharomyces cerevisiae*, *Caenorhabditis elegans*, and *Drosophila melanogaster* [10,11,12], that proteins with high degree (or hubs) in the network are more likely to be essential than those selected by chance [13]. From the topological perspective, the removal of such high-degree nodes makes the network collapse into isolated clusters. From a biological view, a few highly connected proteins generally guarantee the functional integrity of the network. This phenomenon is commonly referred to as the centrality-lethality rule, which was first observed by Jeong and colleagues [13]. The centrality-lethality rule demonstrates a high correlation between a node's topological prominence in a protein interaction network and its essentiality. Since then, much attention has been given to the study of high-degree nodes or hubs in protein interaction networks [11,14,15,16,17,18,19]. Most of the authors confirmed the correlation between degree centrality and protein essentiality [11,14,18,19] and some authors examined the reasons for such a correlation [16,17].

Several other topological properties of nodes, such as betweenness centrality [20,21], closeness centrality [22], subgraph centrality [23], eigenvector centrality [24], and information centrality [25], have also been proposed for the discovery of essential proteins, besides the degree centrality. The use of centrality measures based on network topology has become an important means in the study of essential proteins, which is fundamental in many application [26]. For example, the essential genes in pathogenic organisms can be taken as the potential targets for new antibiotics [27] and the identification of essential genes and non-essential genes is valuable for rational drug design [28].

The current centrality measures treat all edges in the network equally. However, some protein-protein interactions are more important than others in reality [16]. Specially, the protein interaction data generated by high-throughput technologies include high false positives [29,30]. Intuitively, the false positives and the real physical interactions can not be treated equally. In this paper, we propose a new method for evaluating the confidence of each interaction and redefined six standard centrality measures in weighted network. The experimental results show that our consideration is particularly meaningful in the discovery of essential proteins. More essential proteins are detected by the weighted centrality measures than by the original centrality measures used in unweighted network.

2 Centrality Measures and Evaluation

A protein interaction network is represented as an undirected graph $G(V, E)$ with proteins as nodes and interactions as edges. Recently, much attention has

been given for the relationship between the centrality of a protein within an interaction network and its essentiality for the organism's survival. There are several commonly used centrality measures for predicting a protein's essentiality, such as degree centrality (DC)[13], betweenness centrality (BC)[20,21], closeness centrality (CC)[22], subgraph centrality(SC)[23], eigenvector centrality(EC)[24], and information centrality(IC)[25]. These different centrality measures have been described in recent reviews [19,31,32].

Though previous authors have tested these centrality measures in protein interaction networks, they did not take the universal false positives into consideration. However, the protein-protein interaction data obtained from large scale, high-throughput experiments generally contain false positives [29,30]. To evaluate how the false positives affect the discovery of essential proteins, we reassessed systematically the six centrality measures (DC, BC, CC, SC, EC, and IC) by using three data sets with different confidence levels. The three test data sets were from von Mering *et al.* 2002 [29]. To describe simply, we name the three test data Y2k (2455 interactions), Y11k (11000 interactions), and Y45k (45000 interactions), respectively, according to the number of edges included in them. Y2k, which consists of the first 2455 reliable interactions, is a subset of Y11k, and Y11k is a subset of Y45k. A list of essential genes was obtained from the MIPS database[33]. For each centrality measure, we use it to rank all proteins in the network according to their centrality values and select a certain number of the top proteins. Similar to most experiments[19], we also select the top 15%, 20%, and 25% proteins as essential candidates. The *precision*, scored as $TP/(TP+FP)$, is used to validate the essential proteins discovery, where TP, true positives, are true essential proteins that are in the discovery and FP, false positives, are non-essential proteins that are in the prediction. The *precision* of the six centrality measures for uncovering essential proteins in the three test data is shown in Fig.1.

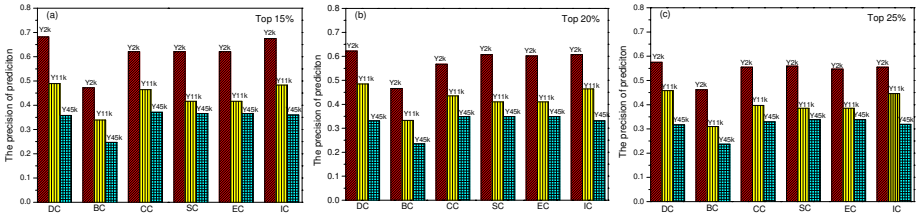


Fig. 1. The precision for essential proteins discovery in three protein interaction networks with different confidence levels. (a)Top 15% proteins are selected, (b)Top 20% proteins are selected, (c)Top 25% proteins are selected.

From Fig.1 we can see that Lower precision is obtained for all centrality measures (DC, BC, CC, SC, EC, and IC) when being used in network of less confidence level to predict the same proportion proteins. As shown in Fig.1 (a), from the high reliable network Y2k to the middle reliable network Y11k, there are

more than 10% decrements. When the unreliable network Y45k is used, all these centrality measures perform on even much less precision. The similar downward trends of precisions for DC, BC, CC, SC, EC, and IC can also be seen in Fig.1 (b) and (c) when the top 20% and 25% proteins are selected. As one can seen in Fig.1, the precision of essential proteins prediction based on network topology depends heavily on the reliability of the network. Therefore, the confidence of the protein-protein interactions should be taken into account when predicting the essential proteins based on network topology. In the following section, we will redefine the six standard centrality measures in weighted protein interaction networks.

3 Definition of Centrality Measures in Weighted Protein Interaction Network

3.1 Definition of Weighted Centrality Measures

A weighted protein interaction network can be represented as a weighted undirected graph $G = (V, E)$. Each edge $(i, j) \in E$ is assigned with a weight $w_{i,j}$, which represents the probability of this interaction between node i and node j being a true positive. To describe simply, the centrality measures (DC[13], BC[20,21], CC[22], SC[23], EC[24], and IC[25]) of a weighted graph G are accordingly marked as DC^W, BC^W, CC^W, SC^W, EC^W, and IC^W.

Definition 1. The weighted degree centrality $DC^W(i)$ of a node i is the sum of weights of the edges connecting node i and its neighbors.

$$DC^W(i) = \sum_{j \in N_i} w_{i,j} \tag{1}$$

where N_i is the set of neighbors of node i.

Definition 2. The weighted betweenness centrality $BC^W(i)$ of a node i is equal to the average fraction of shortest paths that pass through the node i.

$$BC^W(i) = \sum_{s} \sum_{t} \frac{\sigma_{st}(i)}{\sigma_{st}}, \ s \neq t \neq i \tag{2}$$

where σ_{st} denotes the total number of shortest paths between s and t and $\sigma_{st}(i)$ denotes the number of shortest paths from s to t that pass through the node i.

The main difference between $BC^W(i)$ and $BC(i)$ is the calculation of shortest path. In a unweighted graph G, a shortest path between two nodes is a path that has the minimum constituent edges. However, the shorted path may be misdirected with the false positives in protein interaction networks. For example, of all the three paths between node i and node j showed in Fig.2, the path $\{i,v_1,j\}$ will be the choice for the shortest path without regard to the edges' reliability. As a matter of fact, however, the path $\{i,v_4,v_5,v_6,j\}$ is a real pathway from node i to node j. Therefore, we use $c_{i,j} = 1/w_{i,j}$ to describe the cost of edge

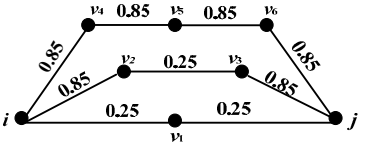


Fig. 2. The shortest path from node i to node j is $\{i,v_4,v_5,v_6,j\}$ with the lowest cost of 4.7

connecting node i to node j. The *cost* of a path is the sum of all the *costs* of its constituent edges. For example, in Fig.2, the *cost* of path $\{i,v_1,j\}$ is 8.0, the *cost* of path $\{i,v_2,v_3,j\}$ is 6.4, the *cost* of path $\{i,v_4,v_5,v_6,j\}$ is 4.7. Obviously, the path $\{i,v_4,v_5,v_6,j\}$ is more likely to be a real pathway with the lowest *cost*. Therefore, the shortest path between a given pair of nodes is the path that has the lowest *cost*. In this paper, we mark the *cost* of a shortest path p from node i to node j as $c_p(i,j)$ for simple and clear description.

Definition 3. The weighted closeness centrality $CC^W(i)$ of a node i in a weighted graph G can be defined as:

$$CC^W(i) = \frac{1}{\sum_{j \neq i} c_p(i,j)} \tag{3}$$

$CC^W(i)$ is a global metric which describes how the given node i connects to other nodes.

Definition 4. The weighted subgraph centrality $SC^W(i)$ of a node i in a weighted graph G can be defined as:

$$SC^W(i) = \sum_{l=0}^{\infty} \frac{\mu_l(i)}{l!} \tag{4}$$

where $\mu_l(i)$ denotes the number of closed walks of length l which starts and ends at node i.

Definition 5. The weighted eigenvector centrality $EC^W(i)$ of a node i in a weighted graph G is defined as the ith component of the principal eigenvector of A, where A is an edge weight matrix. Let λ be an eigenvalue and e be the eigenvector. Then for an equation $\lambda e = Ae$, we can obtain $EC^W(i) = e_1(i)$, where e_1 corresponds to the largest eigenvalue of A.

Definition 6. The weighted information centrality $IC^W(i)$ of a node i in a weighted graph G is defined as:

$$IC^W(i) = [\frac{1}{n} \sum_j \frac{1}{I_{ij}}]^{-1} \tag{5}$$

where n is the number of nodes in graph G and $I_{ij} = (r_{ii} + r_{jj} - r_{ij})^{-1}$. Let D be a diagonal matrix of the weighted degree of each node and J be a matrix with all its elements equal to one. Then, we get $R = (r_{ij}) = [D - A + J]^{-1}$. For computational purposes, I_{ii} is defined as infinite. Thus, $\frac{1}{I_{ii}} = 0$.

3.2 Construct Weighted Protein Interaction Network

To construct weighted protein interaction network, we assign confidence score to each interaction by combining two aspects: (1) observing its experimental evidences; (2) evaluating its function similarity.

For aspect (1), we use the logistic regression-based model employed in [34,35] to examine the reliability of an interaction. For each interaction, its reliability score is determined by its experimental evidences and the number of observations in each experimental type. The experiments are classed into four categories: co-immunoprecipitation screens, yeast two-hybrid assays, large scale experiments and small scale experiments. The reliability score of an interaction between node i and node j is marked as $L_O(i, j)$ for it is computed based on logistic regression.

For aspect (2), we calculate the function similarity of two connected proteins based on GO (Gene Ontology) semantic similarity. There have been several methods for computing the similarity between GO terms [36,37,38]. Here, we select the widely used method proposed by Lin[37], in which the similarity of two GO terms c_1 and c_2 is defined as:

$$sim(c_1, c_2) = \frac{2 \max_{c \in C_T(c_1,c_2)}(log\ p(c))}{log\ p(c_1) + log\ p(c_2)} \tag{6}$$

where $p(c)$ denotes the probability of encountering term c in the target species and $C_T(c_1, c_2)$ denotes the sets of common ancestors of term c_1 and term c_2.

Let F_i and F_j denote the sets of function annotations for protein i and protein j, respectively. Then, the function similarity $S_F(i, j)$ of protein i and protein j can be defined as:

$$S_F(i, j) = max_{c_1 \in F_i, c_2 \in F_j}(sim(c_1, c_2)) \tag{7}$$

To satisfy that the confidence score of each interaction should be between 0 and 1, the following normalization operation is performed.

$$S_F^*(i, j) = \frac{S_F(i, j) - Min\_S_F}{Max\_S_F - Min\_S_F} \tag{8}$$

where Max_S_F and Min_S_F denote the maximum value and minimum value of all the interactions' function similarity scores, respectively.

By considering both the reliability measurement and the function similarity, we define the confidence score $C_{score}(i, j)$ of an interaction connecting protein i and protein j as formula (9):

$$C_{score}(i, j) = \frac{L_O(i, j) + S_F^*(i, j)}{2} \tag{9}$$

4 Results

To evaluate whether our method for evaluating the confidences of protein-protein interactions works and to investigate whether the definitions of centrality measures in weighted protein interaction network outperform that in unweighted network, we implement them to predict the essentiality of proteins from *Saccharomyces cerevisiae* for its well characterized by knockout experiments and widely used in previous works[19]. The protein-protein interactions of *Saccharomyces cerevisiae* are downloaded from DIP database[39]. The interaction map is considered as a network in which proteins are represented as nodes and interactions connecting two proteins are represented as edges. There are 4746 proteins and 15166 interactions in total without self-interactions and repeated interactions. For the functional annotation of the proteins, we use functions classified as molecular function by GO[40]. The lethal proteins are obtained from MIPS[33].

Proteins are ranked according to their values of centrality measures and a certain top percentage of proteins are selected as candidates for essential proteins. Then we determine how many of them are essential. For evaluation, we compare the results of centrality measures in weighted network against those in unweighted network. The comparison results are shown in Fig.3.

In Fig.3 we illustrate the number of essential proteins identified by DC^W, BC^W, CC^W, SC^W, EC^W and IC^W as well as by DC, BC, CC, SC, EC and IC with the top 10%, 15%, 20% and 25% of proteins in the protein interaction network. From Fig.3 we can see that all the weighted centrality measures perform significantly better than the unweighted centrality measures in the selection of essential proteins in the yeast protein interaction network. Especially, the improvements of SC^W and CC^W are remarkable for that about 20% extra essential proteins are detected by SC^W and CC^W than by SC and CC. The results also indicate that all the centrality measures based on topological characters are sensitive to the interactions' confidence, which in turn illustrate that effective weighting methods are important.

Take the simplest method DC for example, low-connectivity proteins negatived by it may be essential. On the contrary, non-essential proteins with high interactions may be false predicted. In Fig.4, we give three examples of essential proteins: YDR356W, YNL216W, and YBR060C with 8 interactions. Their weighted degrees are all larger than 4.5. By contrast, the three examples shown in Fig.5 which have more than 20 interactions are validated to be not essential. Their weighted degrees are smaller than 4.0. Of course, we can not say that DC^W can identify all the essential proteins negatived by DC and filtered all the non-essential proteins false predicted by DC. However, more essential proteins are discovered by DC^W than by DC. This phenomenon is more obvious for SC^W and SC, CC^W and CC, BC^W and BC as they are all based on the calculation of path length. And, false positives affect heavily on the calculation of path length as we have discussed previously.

To evaluate all the centrality measures more generally, we cite the jackknifing methodology developed by Holman *et al.*[41]. As shown in Fig.6, proteins are ordered by highest to lowest values of centrality measures and the cumulative count

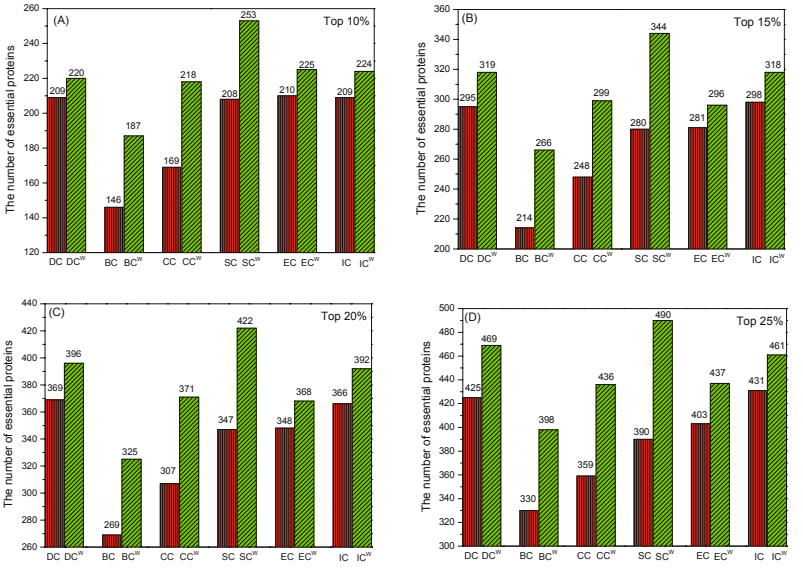


Fig. 3. Comparison of the number of essential proteins that selected from unweighted network and that selected from weighted network by ranking proteins according to their values of centrality measures

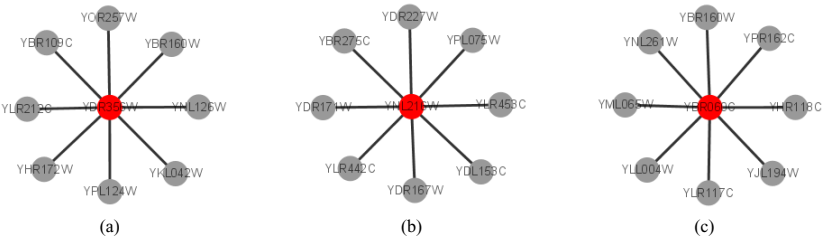


Fig. 4. Examples of low-connectivity essential proteins

of essential proteins is plotted. The area under the curve (AUC) for the weighted centrality measures and that for the unweighted methods are compared. In addition, an ideal ranking is plotted with all essential proteins artificially placed at the beginning of the list. Moreover, 10 random assortments are also plotted for comparison. As can be seen in Fig.6, of all the weighted centrality measures the sorted curves appear to be better than their corresponded unweighted curves. More over, all the sorted curves of centrality measures appear well differentiated from the randomized sorting, which indicates that the discovery of essential proteins based on topological characters are statistical significance.

The above analysis demonstrates that the weighted centrality measures are effective in predicting protein essentiality even false positives existed in the protein

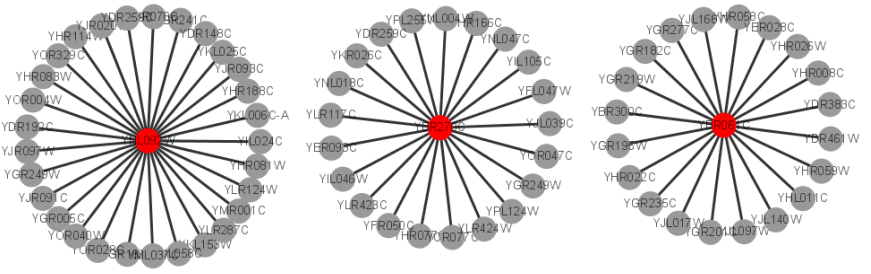


Fig. 5. Examples of high-connectivity non-essential proteins

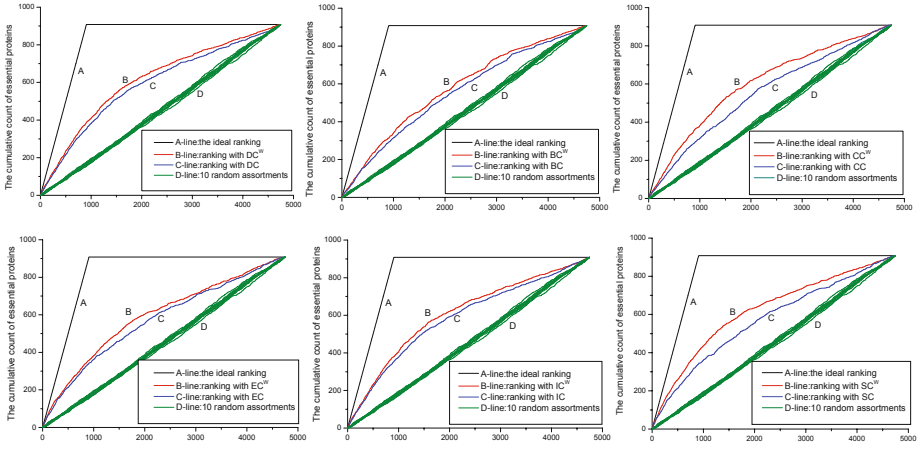


Fig. 6. Essential proteins discovery is validated by a jackknife methodology. The X-axis represents the ranked proteins in the yeast protein interaction network, ranked from left to right as highest to lowest values of centrality measures. The Y-axis is the cumulative count of essential proteins with respect to the ranked proteins moving left to right. Line A is the ideal ranking. Line B is the sorting by weighted centrality measures. Line C is the ranking by unweighted centrality measures. Line D are 10 random assortments.

interaction network. Then, one question is whether the weighted centrality measures are also valid in high confidence protein-protein interactions. We get the top 5283 high reliability interactions from DIP [39] as Core dataset. There are 2373 proteins in total, of which 666 proteins are essential. In Table 1, a comparison of the precision of the weighted centrality measures: DC^W, BC^W, CC^W, SC^W, EC^W and IC^W, and that of the unweighted ones: DC, BC, CC, SC, EC and IC is shown both in the original protein interaction network (All PPIs) and the cleaned network (Core PPIs). The comparison result shows that the weighted centrality measures perform well both in the high positive protein interaction network and the high confidence network.

Table 1. The precision of essential protein discovery on original protein interaction network (All PPIs) and the cleaned network (Core PPIs)

		DC^W	DC	BC^W	BC	CC^W	CC	SC^W	SC	EC^W	EC	IC^W	IC
All PPIs	Top 10%	0.46	0.44	0.39	0.36	0.46	0.36	0.53	0.44	0.47	0.44	0.47	0.44
	Top 15%	0.45	0.42	0.37	0.34	0.42	0.35	0.48	0.39	0.42	0.39	0.45	0.42
	Top 20%	0.42	0.39	0.34	0.32	0.39	0.32	0.44	0.37	0.39	0.37	0.41	0.39
	Top 25%	0.40	0.37	0.34	0.30	0.37	0.30	0.41	0.33	0.37	0.34	0.39	0.36
Core PPIs	Top 10%	0.50	0.50	0.44	0.42	0.42	0.39	0.55	0.52	0.53	0.42	0.47	0.47
	Top 15%	0.49	0.47	0.41	0.40	0.40	0.37	0.50	0.47	0.47	0.39	0.45	0.45
	Top 20%	0.47	0.47	0.41	0.39	0.41	0.36	0.48	0.43	0.44	0.37	0.44	0.41
	Top 25%	0.44	0.43	0.39	0.37	0.40	0.36	0.46	0.41	0.43	0.36	0.42	0.38

5 Conclusions and Future Work

By assessing the performance of six standard centrality measures for the discovery of essential proteins in different confidence-level networks, we find that the centrality measures based on network topology are very sensitive to false positives. Thus, we introduce a new weighing method for evaluating the confidence of protein-protein interactions. The experimental evidences are considered by using the logistic regression-based model. And function similarity between proteins are also considered in our method. Based on the new weighting method, we construct a weighted protein interaction network and redefine the six standard centrality measures in weighted network. The experimental results show that: (1)the weighting method can improve the performance of centrality measures considerably; (2)the weighted centrality measures are significantly to be better than the original centrality measures used in unweighted network, Even about 20% improvements are obtained from closeness centrality and subgraph centrality; (3)the weighted centrality measures perform well both in the high positive networks and high confidence networks; (4)the prediction based on network topology is significantly better than random selection.

The proposed weighing method can also be used in other fields, such as identification of protein complexes and functional modules. As future work, it would be interesting to apply this weighing method to other studies. Moreover, analysis of the relation among different centrality measures and exploitation of an ensemble approach for integrating these different types of prediction methodologies are also our future work.

References

1. Uetz, P., et al.: A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. Nature 403, 623–627 (2000)
2. Barabasi, A.L., Oltvai, Z.N.: Network biology: understanding the cell's functional organization. Nat. Res. 5, 101–114 (2004)

3. Yook, S., Oltvai, Z., Barabasi, A.: Functional and topological characterization of protein interaction networks. Proteomics 4, 928–942 (2004)
4. Watts, D.J., Strogatz, S.H.: Collective dynamics of small-world networks. Nature 393, 440–442 (1998)
5. Rives, A.W., Galitski, T.: Modular organization of cellular networks. Proc. Natl. Acad. Sci. 100, 1128–1133 (2003)
6. Gavin, A.C., et al.: Proteome survey reveals modularity of the yeast cell machinery. Nature 440(7084), 631–636 (2006)
7. Maslov, S., Sneppen, K.: Specificity and stability in topology of protein networks. Science 296, 910–913 (2002)
8. Winzeler, E.A., Shoemaker, D.D., Astromoff, A., Liang, H., Anderson, K., et al.: Functional characterization of the S. cerevisiae genome by gene deletion and parallel analysis. Science 285, 901–906 (1999)
9. Kamath, R.S., Fraser, A.G., Dong, Y., Poulin, G., Durbin, R., et al.: Systematic functional analysis of the Caenorhabditis elegans genome using RNAi. Nature 421, 231–237 (2003)
10. Yu, H., Greenbaum, D., Lu, H.X., Zhu, Z., Gerstein, M.: Genomic analysis of essentiality within protein networks. Trends Genet. 20, 227–231 (2004)
11. Hahn, M. W., Kern, A.: Comparative genomics of centrality and essentiality in three eukaryotic protein-interaction networks. Mol. Biol. Evol. 22(4), 803–806 (2005)
12. Wuchty, S.: Interaction and domain networks of yeast. Proteomics 2, 1715–1723 (2002)
13. Jeong, H., Mason, S., Barabási, A., Oltvai, Z.: Lethality and centrality in protein networks. Nature 411, 41–42 (2001)
14. Batada, N.N., Hurst, L.D., Tyers, M.: Evolutionary and physiological importance of hub proteins. PLoS Comput. Biol. 2(7), e88 (2006) doi:10.1371/ journal.pcbi.0020088
15. Coulomb, S., Bauer, M., Bernard, D., Marsolier-Kergoat, M.: Gene essentiality and the topology of protein interaction networks. Proc. R. Soc. B 272, 1721–1725 (2005)
16. He, X.L., Zhang, J.Z.: Why Do Hubs Tend to Be Essential in Protein Networks? PLoS Genetics 2(6), 826–834 (2006)
17. Zotenko, E., Mestre, J., O'Leary, D.P., Przytycka, T.M.: Why Do Hubs in the Yeast Protein Interaction Network Tend To Be Essential: Reexamining the Connection between the Network Topology and Essentiality. PLoS Computational Biology 4(8), 1–16 (2008)
18. Vallabhajosyula, R., Chakravarti, D., Lutfeali, S., Ray, A., Raval, A.: Identifying Hubs in Protein Interaction Networks. Plos One 4(4), 1–10 (2009)
19. Ernesto, E.: Virtual identification of essential proteins within the protein interaction network of yeast (2005) http://arxiv.org/abs/q-bio.MN/0505007
20. Narayanan, S.: The betweenness centrality of biological networks. Master of Science in Computer Science. Virginia Polytechnic Institute and State University (September 16, 2005)
21. Joy, M., et al.: High-betweenness proteins in the yeast protein interaction network. Journal of Biomedicine and Biotechnology 2, 96–103 (2005)
22. Wuchty, S., Stadler, P.: Centers of complex networks. Journal of Theoretical Biology 223, 45–53 (2003)
23. Estrada, E., Rodríguez-Velázquez, J.A.: Subgraph centrality in complex networks. Phys. Rev. E. 71(5) (2005)

24. Bonacich, P.F.: Power and centrality: A family of measures. American Journal of Sociology 92(5), 1170–1182 (1987)
25. Stevenson, K., Zelen, M.: Rethinking centrality: Methods and examples. Social Networks 11, 1–37 (1989)
26. Gerdes, S., Edwards, R., Kubal, M., Fonstein, M., Stevens, R., Osterman, A.: Essential genes on metabolic maps. Curr. Opin. Biotechnol. 17, 448–456 (2006)
27. 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)
28. Lamichhane, G., Zignol, M., Blades, N.J., Geiman, D.E., Dougherty, A., Grosset, J., Broman, K.W., Bishai, W.R.: A postgenomic method for predicting essential genes at subsaturation levels of mutagenesis: application to Mycobacterium tuberculosis. PNAS 100(12), 7213–7218 (2003)
29. von Mering, C., et al.: Comparative assessment of large-scale data sets of protein-protein interactions. Nature 417(6887), 399–403 (2002)
30. Brohee, S., van Helden, J.: Evaluation of clustering algorithms for protein-protein interaction networks. BMC Bioinformatics, 7–488 (2006)
31. Jacob, R., Koschtzki, D., Lehmann, K.A., Peeters, L., Tenfelde-Podehl, D.: Algorithms for Centrality Indices. In: Brandes, U., Erlebach, T. (eds.) Network Analysis. LNCS, vol. 3418, pp. 62–82. Springer, Heidelberg (2005)
32. Mason, O., Verwoerd, M.: Graph theory and networks in biology. IET Systems Biology 1(2), 89–119 (2006)
33. Mewes, H.W., Amid, C., Arnold, R., et al.: MIPS: analysis and annotation of proteins from whole genomes. Nucleic Acids Research 32, D41–D44 (2004)
34. Sharan, R., et al.: Conserved patterns of protein interaction in multiple species. PNAS 102(6), 1974–1979 (2005)
35. Shlomi, T., Segal, D., Ruppin, E., Sharan, R.: Qpath: a method for querying pathways in a protein-protein interaction network. BMC Bioinformatics, 7–199 (2006)
36. Resnik, P.: Using information content to evaluate semantic similarity in taxonomy. In: Proc. the 14th International Joint Conference on Artificial Intelligence, pp. 448–453 (1995)
37. Lin, D.: An information-theoretic definition of similarity. In: Proc.of 15th International Conference on Machine Learning, pp. 296–304 (1998)
38. Lei, Z., Dai, Y.: Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction. BMC Bioinformatics 7, 491 (2006)
39. http://dip.doe-mbi.ucla.edu/
40. Issel-Tarver, L., Christie, K.R., Dolinski, K., et al.: Saccharomyces Genome Database. Methods Enzymol. 350, 329–346 (2002)
41. Holman, A.G., Davis, P., Foster, J.M., et al.: Computational prediction of essential genes in an unculturable endosymbiotic bacterium, Wolbachia of Brugia malayi. BMC Microbiology 9, 243 (2009)

Identifying Differentially Abundant Metabolic Pathways in Metagenomic Datasets

Bo Liu and Mihai Pop

Center for Bioinformatics and Computational Biology,
Institute for Advanced Computer Studies,
Department of Computer Science,
Univeristy of Maryland, College Park, USA
{boliu,mpop}@umiacs.umd.edu

Abstract. Enabled by rapid advances in sequencing technology, metagenomic studies aim to characterize entire communities of microbes bypassing the need for culturing individual bacterial members. One major goal of such studies is to identify specific functional adaptations of microbial communities to their habitats. Here we describe a powerful analytical method (MetaPath) that can identify differentially abundant pathways in metagenomic data-sets, relying on a combination of metagenomic sequence data and prior metabolic pathway knowledge. We show that MetaPath outperforms other common approaches when evaluated on simulated datasets. We also demonstrate the power of our methods in analyzing two, publicly available, metagenomic datasets: a comparison of the gut microbiome of obese and lean twins; and a comparison of the gut microbiome of infant and adult subjects. We demonstrate that the subpathways identified by our method provide valuable insights into the biological activities of the microbiome.

Keywords: Metagenomics; Metabolic Pathway.

1 Introduction

Metagenomics is a new scientific field that involves the analysis of organismal DNA sequences obtained directly from an environmental sample, enabling studies of microorganisms that are not easily cultured in a laboratory [1]. Metagenomic studies, pioneered in the early 2000s [2], have recently increased in number and scope due to the emergence of next generation sequencing technologies. Due to the difficulty of assembling entire organisms from a metagenomic data-set, most analyses take a gene-centric view, treating the community as an aggregate and ignoring the exact assignment of genes to individual organisms. In fact, it can be argued that the environment is better characterized by its gene complement than by its taxonomic composition, given that similar biological functions can be performed by microbes of distinct taxonomic origins [3]. The functional profile for a sample can be recovered by mapping sequences to gene families [4], subsystems [5] or metabolic pathways [6]. The relative abundance of

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 101–112, 2010.

each functional category can be estimated by counting how many sequences are assigned to each category, and this information is the basis for detailed comparisons of the functional potential of different functions. In a typical comparative metagenomics experiment, sequences are generated from a collection of samples belonging to two groups, for example, obese or lean twins [3], and healthy infants or adults [7]. An important biological problem is to find differentially abundant functional signatures (e.g., genes or metabolic pathways) that are selected for by their local environments. Traditional analysis compares the relative abundances of the categories one-at-a-time between different phenotypes, and computes the significance using one of several statistical approaches [8,9,10]. When comparing communities at the gene family level, many functional categories are commonly found to be differentially abundant, even after correcting for multiple hypothesis testing [3,7]. The interpretation of these data can be daunting. An alternative approach focuses on functional subsystems and metabolic pathway comparisons [11], the number of which is much smaller than gene families. Results at these levels are easier to interpret and can provide a stronger evidence of distinct functional capacities than at the level of individual gene families. Such analyses, however, can be unnecessarily coarse. For example, the use of KEGG pathways as a basis for analysis is complicated by the following issues: (1) the definitions of pathways in KEGG are coercive, and the interactions between these pathways are ignored; (2) the genes in a pathway may not be fully covered by the identified genes in a metagenomic sample; (3) significant differences in the abundance of certain genes may be masked once the abundance of all genes in a pathway is aggregated.

To address these problems, we introduce a general method (MetaPath) for searching the global metabolic pathway to find differentially abundant finer-level subpathways. For the purposes of this paper we define a subpathway to be a connected set of genes that is statistically enriched or depleted in one group of samples. Underlying our approach is a statistical scoring system that captures the differential abundance for a given subpathway, combined with a search algorithm, based on a maximum weighted subgraph heuristic, for indentifying the highest scoring subpathways. Unlike previous approaches, MetaPath explicitly searches significant subpathways in the global metabolic pathway (rather than the KEGG-defined pathways), enabling us to detect subpathways spanning predefined containers. In addition, we developed rigorous statistical methods that take into account the topology of the network when testing the significance of the subpathways. Using simulated data-sets, we demonstrate that Metapath outperforms previously described approaches for comparing biological networks based on abundance data. We show that our findings are more robust to noisy data than the results of single gene comparisons, and that MetaPath can find finer-level subpathways than can be found by comparing predefined KEGG pathways. We also discuss the biological significance of the results derived from the application of MetaPath to actual metagenomic data-sets, demonstrating that the output from MetaPath is easy to interpret and provides valuable biological insights. The software is freely available at `http://cbcb.umd.edu/~boliu/metapath/`

2 Methods and Datasets

2.1 Datasets

We test our methods on two previously published metagenomic datasets, which were downloaded from the NCBI Trace Archive or Short Read Archive databases: (1) Gut Microbiomes from Obese and Lean Twins [3]; (2) Metagenomes from Adult- and Infant-Type Gut Microbiomes [7]. Each dataset is divided into two populations of distinct phenotypes. The metabolic pathway data were downloaded from the KEGG pathways database [6]. The metabolic information is represented as a graph where nodes are metabolic substrates, and edges are molecular reactions (Fig. 1). The edges could be unidirectional or bidirectional depending on whether the corresponding reaction is reversible. Multiple reactions related to a same biological process are aggregated by KEGG into a pathway. In addition, we refer to the graph comprising all reactions in KEGG as the global metabolic pathway. Metagenomic sequences are annotated through BLASTX searches against KEGG genes database. The abundance of each molecular reaction is estimated as the number of metagenomic sequences mapped to it. Note that more accurate abundance estimates can be obtained by taking into account the length of individual genes [12] and we plan to explore the use of such estimates (and the associated statistics) in future versions of our software.

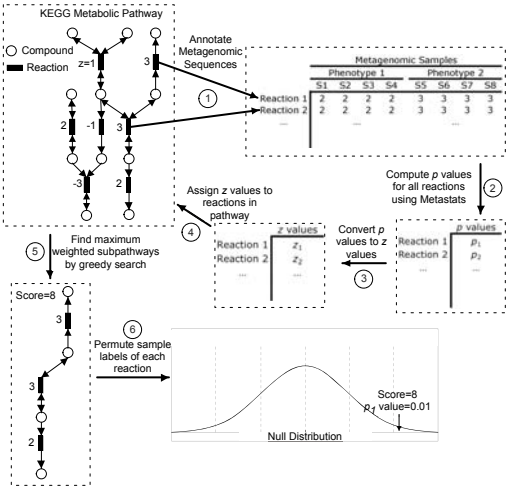


Fig. 1. Schematic diagram of our methods. Sequences from each sample are annotated against KEGG database and mapped to reactions in pathways leading to an abundance matrix where the rows are different reactions and columns are samples. Then p values are computed for all reactions using Metastats [9], converted into Z values, then greedy search is performed on the edge-weighted graph to find maximum weighted subpathways. Finally, we estimate the null distribution of the subpathway score by randomly permuting the sample labels, and compute the corresponding p values.

2.2 Scoring Metabolic Subpathways

To score the biological activity of a particular subpathway, we first use Metastats [9] to calculate the significance of differential abundance for each reaction between two groups. Under the null hypothesis, the relative abundances are random and have the same distribution across phenotypes, thus the p values follow a uniform distribution from 0 to 1. Based on this assumption, p values can be converted to Z scores [13]. Because Metastats performs a two-tailed test for each reaction, the two-tailed p values can be converted back to the original Z values using the following equation:

$$Z_i = \begin{cases} CDF_{sn}^{-1}(1 - p_i/2) \times -1 \text{ , if mean(G1)} < \text{mean(G2)} \\ CDF_{sn}^{-1}(1 - p_i/2) \qquad \text{ , if mean(G1)} > \text{mean(G2)} \end{cases} \tag{1}$$

$CDF_{sn}^{-1}$is the inverse cumulative density function of standard normal distribution; G1 and G2 represent populations 1 and 2. Using this formula, if a reaction is more abundant in population G1, then its Z score will be positive and vice versa. We are specifically interested in finding a pathway whose reactions are either enriched or depleted as a whole, as apposed to previous approaches [13] [14] that identify active or perturbed subnetworks, which may contain a mixture of enriched and depleted components. We define the aggregate score for a particular subpathway to be the sum of the Z scores over all reactions contained within it: $Z = \frac{1}{\sqrt{k}} \sum_{1,k} Z_i$.

We attempt to find pathways that maximize the cumulative Z-score defined above. Unfortunately, this problem is NP-hard, equivalent to finding a maximum-weight subgraph [13]. Several approaches to solving this problem have been previously proposed: [13] used simulated annealing, but this heuristic is slow; [14] used integer linear programming that can find provably optimal subpathways quickly, but it requires the commercial software CPLEX which is not available to the general public (re-coding this algorithm using other freely available ILP solvers is beyond the scope of this paper. Here we rely on a greedy heuristic that is fast, and, while not guaranteed to find maximally scoring pathways, performs well in practice. We restrict our search to pathways of a fixed size k, in order to enable the computation of the statistical significance of pathways. This restriction enables us to more accurately compute the null distribution of pathway scores that is highly dependent on the size. The algorithm we use:

Input: A metabolic pathway graph G=(V, E), where V and E are reactions; a set of weight values Z associated with each edge in graph G; a number k which determines the size of subpathway.
Output: A subpathway G_{max} of G and its score W_{max}.
Initialize W_{max} to 0;
for all edge e_i in E **do**
  Initialize G_{tem} by including e_j;
  for $i = 1$ to k **do**
    Pick e_j which has the highest weight for all edges adjacent to G_{tem};
  end for

Include e_j to G_{tem};
Calculate the score W_{tem} of G_{tem};
if $W_{tem} > W_{max}$ **then**
$W_{max} = W_{tem}$;
$G_{max} = G_{tem}$;
end if
end for
Output G_{max} and its score W_{max}

This algorithm tries to find a connected subpathway with k edges, which can have any arbitrary structure. However, it is believed that in metabolic pathway, chains are especially more biologically meaningful and interesting, because they attempt to capture the structure of a series of reactions that are successively connected. To allow this idea, we modify line 5 of the above algorithm to Pick an edge e_j which has the highest weight of the edges that are adjacent to and have the same direction with e_{j-1}. Both searching algorithms are implemented in our program. In addition, we also compute the top m high-scoring pathways by iteratively removing the edges in the graph associated with pathways already considered by our algorithm.

2.3 Computing the Significance of Subpathways

The null score distribution for a specific subpathway can be estimated by permuting the sample labels for the reactions and computing the scores of the permuted subpathways. The significance p value is estimated as the number of random permutations that produce higher scores than subpathways in the original dataset. The p-value computed through this approach (termed p_1 throughout the rest of the paper), however, ignores the topology of the underlying network, potentially leading to incorrect conclusions. For example, assume every edge is connected with all other edges in the global metabolic pathway. The best subpathway of size k is simply composed of the top k significant edges. This means whenever there are significant reactions, which may simply come from noise, they will form a significant subpathway. To address this problem, we compute another p value (termed p_2), relying on a topological definition of the null distribution of subpathway scores. Specifically, instead of treating each pathway as a bag of genes, we estimate the distribution of scores for actual pathways identified within the underlying metabolic network. As before, this null-distribution depends on the length k of a pathway. The p_2 value is calculated as follows for a subpathway of size k: (i) Permute the reaction scores (row labels of the abundance matrix) of the global metabolic pathway (Fig. 1). (ii) Perform greedy search to find the maximum weighted subpathway of size k. (iii) Repeat step 1 and step 2 for 1000 times, and generate 1000 scores (null distribution). (iv) The p_2 value is the proportion that we see scores higher than Z. The algorithm described above is parameterized by the size k of pathways of interest. MetaPath runs this algorithm for values of k within a user-defined range (default 2-10). Redundancy in the pathways reported by this algorithm is eliminated by post-processing the results and eliminating any pathway that is fully contained in a longer pathway.

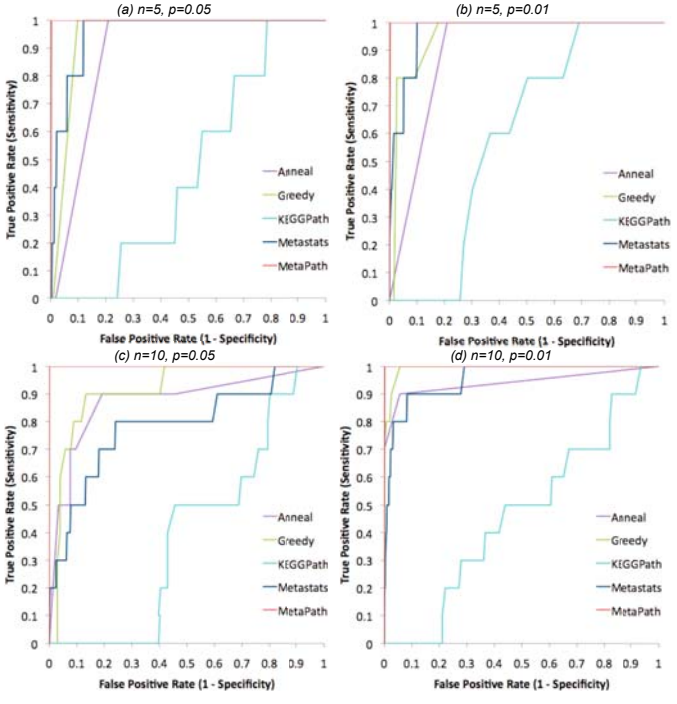


Fig. 2. Comparison of statistical methods of discovering significant reactions in simulated datasets. Four methods are evaluated: discovering active subnetworks using simulated annealing (Anneal) and greedy search (Greedy) [13], discovering significant individual reactions using Metastats [9], finding differentially abundant KEGG-defined pathways (KEGGPath), and MetaPath. Four datasets are created by varying the number of significant reactions n and their significance value p.

3 Results and Discussions

3.1 Performance Evaluation Using Simulated Datasets

In order to validate our methods, we have designed a simulated metagenomic study and compared the results with three previous approaches: (i) identifying active subnetworks using simulated annealing and greedy search [13]; (ii) discovering significant individual reactions using Metastats [9]; and (iii) finding differentially abundant KEGG defined pathways, an approach widely used in metagenomic functional comparison [3,7,10]. We choose these tools because they are addressing similar biological problems. However they do not exactly solve the problem in this paper, which is finding differentially abundant pathways. Here the goal of this simulated study is to show that our problem can not be solved by directly applying methods previously developed in a related context. We designed a simulated metagenomic study in which five subjects are created for each of the two groups with distinct phenotypes. To generate the artificial

reaction abundance matrix (where rows represent reactions and columns represent subjects), for each reaction a normal distribution is created, whose mean is randomly chosen from real metagenomic datasets (obese and lean twins in our study). The variance is calculated by setting the relative standard deviation (standard deviation divided by the mean) to 0.2. If we define a reaction to be equally abundant between two populations, then a random abundance value is generated from the same normal distribution for each subject. Otherwise, if a reaction is defined to be significantly enriched in one population, then another normal distribution is created for this reaction by increasing the mean such that the p value of the t-test for the two distributions is less than a predefined value (0.05 and 0.01 were used in our study). In this study, we have chosen a series of reactions (length 5 or 10) to be enriched in one population. The goal is to compare different methods in recovering these significant reactions based on the simulated abundance matrix. Biologically, the enriched pathways indicate functional enrichment of certain biological processes in a microbial community.

The receiver operating characteristic (ROC) curve is plotted for each method (Fig. 2). Fig. 2 shows that MetaPath outperforms all other methods dramatically showing the advantage in finding small significant subpathways. The most commonly used approach – comparing KEGG defined pathways – performs the worst in our simulation study (Fig. 2).

3.2 Obese and Lean Twins

We used MetaPath to compare the functional potential of the gut microbiome of lean and obese subjects relying on data from [3]. This metagenomic dataset comprises 6 samples from obese subjects and 6 samples from lean objects. The sequences are annotated and mapped to KEGG reactions using BLASTX (E value $<$ 10-5, bitscore $>$ 50, and %identity $>$ 50), resulting in total 1832 unique reactions within the 12 metagenomic samples. First, we computed p values and q values using Metastats to find differentially abundant reactions. Using a p value cutoff of 0.05, 92.7±9.1 (meanstandard deviation) reactions are significant including 37.1±6.6 and 55.6±3.1 enriched reactions in obese and lean groups, respectively, based on 10 runs of Metastats. The high variance of the number of significant genes can be primarily explained by two reasons: (1) some reactions are slightly below or above significance (0.05), thus p values computed through bootstrapping will jump between being considered significant and nonsignificant; (2) large variances of the abundance values within individuals in a same phenotypic group. The q values for all reactions are 1 (except R01676 where q=0.73), which can be explained by the flat distribution of the p values (very few true significant genes), from which the q values are estimated. This is one limitation of relying on the false discovery rate, which requires the estimation of the proportion of features that are truly null [15], approach that does not perform well when only few features are truly significant.

We, then, used MetaPath to search for significant subpathways whose sizes are between 2 and 10, and have found 10 differentially abundant subpathways (Fig. 3) (0.05 cutoff for both p_1 and p_2). All these reactions are enriched in obese

subjects. No subpathway was found to be enriched in lean subjects. These 10 significant subpathways contain 50 unique reactions, 24 of which are significant. It is worth pointing out that the number of significant reactions varies between different runs of statistical permutations (using Metastats) as shown above, but the significant pathways identified by Metapath stay the same 3. This observation confirms that the results from MetaPath are more robust in the presence of noise in the data than the gene-by-gene approach. Five subpathways (Fig. 3a-3e) are completely contained in the Fatty Acid Biosynthesis pathway, which consists of catabolic processes that can generate energy and primary metabolites from fatty acids. Our findings are consistent with previous observations in biochemical analysis and microbiota transplantation experiments in germ-free mice [16], where the concentrations of short-chain fatty acids in the caeca of obese mice are higher than lean mice, suggesting that the gut microbiome in obese subjects has an increased capacity for dietary energy harvest.

Another interesting significant pathway consists of 10 reactions (Fig. 3f), of which 8 belong to Cysteine and Methionine Metabolism and 2 belong to Sulfur Metabolism. Many reactions in this subpathway are connected by the molecule L-Homocysteine. In addition, three other subpathways (Fig. 3g-3i) we discovered

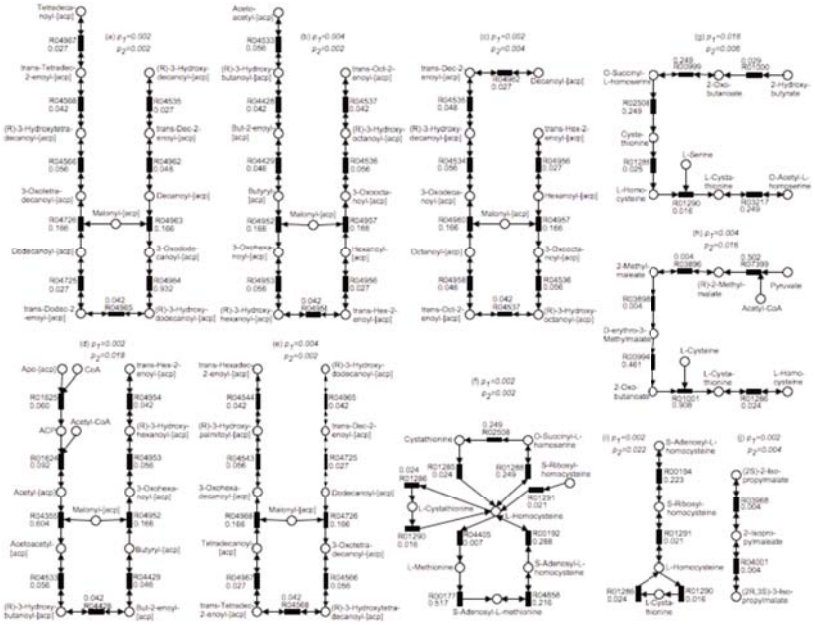


Fig. 3. Comparison of statistical methods of discovering significant reactions in simulated datasets. Four methods are evaluated: discovering active subnetworks using simulated annealing (Anneal) and greedy search (Greedy) [13], discovering significant individual reactions using Metastats [9], finding differentially abundant KEGG-defined pathways (KEGGPath), and MetaPath. Four datasets are created by varying the number of significant reactions n and their significance value p.

further confirm its potential involvement in obesity, because all these three pathways contain L-homocysteine as metabolite. It is well-known that a high level of blood serum homocysteine is a risk factor for cardiovascular disease [17], and obesity an increasingly prevalent metabolic disorder is closely associated with heart disease [18]. Significant correlations between plasma homocysteine concentrations and obesity have been previously reported [17,19,20,21,22]. The finding of increased potential for homocystein biosynthesis within the obese gut microbiome provides an interesting hypothesis for future studies: that the gut microbiome may either have a direct role in the elevation of homocysteine levels in plasma, or may indirectly affect the hepatic biosynthesis of this amino-acid in the human body.

3.3 Infant and Adult Individuals

A second data-set comprises gut microbiome samples from 4 infants and 9 adults individuals which were sequenced by [7]. The sequences were annotated and mapped to the reactions of KEGG pathway using BLASTX (E value $<$ 10-8, hit length coverage $\geq$ 50% of a query sequence), resulting in total 1781 unique reactions. Based on 10 runs of Metastats, 383.7$\pm$1.56 reactions are significant using p value cutoff of 0.05 and 167.2$\pm$2.7 reactions are significant using a q value cutoff of 0.05.

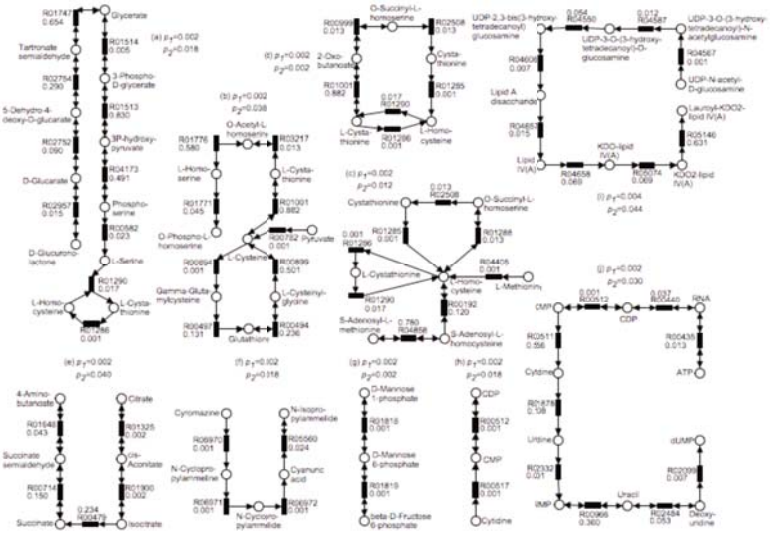


Fig. 4. 10 statistically significant subpathways are found in the infant and adult individuals dataset. 6 subpathways are enriched in the infant subjects (a-f), and 4 subpathways are enriched in the adult subjects (g-j). p_1 and p_2 significance values are shown above each pathway. p value for each reaction is shown with the KEGG reaction number.

Applying MetaPath to search for significant subpathways, we have found 6 subpathways (Fig. 4a-4f) enriched in infant subjects and 4 subpathways (Fig. 4g-4j) enriched in adult subjects. These 10 significant subpathways contain 55 unique reactions, including 38 significant reactions and 17 reactions not found significant by Metastats. Three subpathways (Fig. 4a,c,d) enriched in infant subjects involve the metabolite L-homocysteine, which is consistent with previous observation that breastfed babies have an higher plasma homocysteine level possibly caused by suboptimal availability of folate in breast milk [23]. The concentration of folate is negatively correlated with that of homocysteine, as folate is a necessary coenzyme for reactions that metabolize homocysteine. In addition, babies normally have high protein diet, which may also cause the concentration of homocysteine to increase. A second pathway in Fig. 4e involves substrates citrate and succinate, and is closely related with oxidative tricarboxylic acid (TCA) cycle. TCA cycle is part of carbohydrate metabolism and can convert carbohydrates into usable energy in aerobic organisms. Because the adult gut ecosystem is dominated by strict anaerobes, it is reasonable to find this subpathway enriched in infant individuals where the gut microbiota also includes aerobes. This finding is consistent with results obtained by comparing COG functional categories [7]. We also find a subpathway Fig. 4f belonging to atrazine metabolism to be enriched in infant subjects. Atrazine is one of the most widely used herbicides, and it contaminates water and soil throughout the world. Our finding possibly indicates a side-effect of this contamination.

The pathway in Fig. 4i (enriched in adult subjects) is part of the lipopolysaccharide biosynthesis. Lipopolysaccharides are a building block of the outer membrane of Gram-negative bacteria. The enrichment of pathway Fig. 4i in adult subject may be a result of the fact that Gram-negative bacteria are also enriched in adults. Specifically, Bacteroides, a genus of Gram-negative bacteria, are a major constituent of adult gut microbiome, but not highly prevalent in infants. Fig. 4h and Fig. 4j (enriched in adult) are pathways related with pyrimidine metabolism. The metabolites RNA, cytidine and uridine, which are contained in pyrimidine metabolism, are normally obtained from high RNA food such as organ meats, broccoli, and brewers yeast, which are not available to unweaned infants, as they are not present in high abundance in milk. The pathway in Fig. 4g (enriched in adult) is part of fructose and mannose metabolism a pathway related to carbohydrate metabolism. This is also consistent with COG-based analyses indicating that many mono- or disaccharides metabolism genes are enriched in adults [7], explained by the fact that colonic microbiota in adults uses indigestible polysaccharides as resources for energy production and biosynthesis of cellular components.

4 Conclusions

We have introduced a statistical method for finding significant metabolic subpathways from metagenomic datasets. Compared with previous methods, results from MetaPath are more robust against noise in the data, and have significantly

higher sensitivity and specificity (when tested on simulated datasets). When applied to two publicly available metagenomic data-sets the output of MetaPath is consistent with previous observations and also provides several new insights into the metabolic activity of the gut microbiome. While showing promising results, our methods have several limitations that we plan to address in the near future. First, and foremost, we restrict ourselves to pathways of a fixed length – a restriction necessary for accurately computing the null distribution of pathway scores. This can severely affect our ability to discover long pathways whose abundance differs only slightly, but significantly, between samples. Second, we currently estimate gene abundances by simply counting the number of sequencing reads that map to a certain gene. Such an approach ignores differences in the length of genes, potentially leading to incorrect conclusions. We plan to address this issue by incorporating a recently-published [12] method that can accurately correct for gene-length effects.

Acknowledgments. We thank Niranjan Nagarajan, Carl Kingsford, James White and Saket Navlakha, Theodore Gibbons for helpful discussions. This work was supported in part by grants R01-HG004885 from the NIH, and IIS-0812111 from the NSF, both to MP.

References

1. Riesenfeld, C.S., Schloss, P.D., Handelsman, J.: Metagenomics: genomic analysis of microbial communities. Annual review of genetics 38, 525–552 (2004)
2. Beja, O., Aravind, L., Koonin, E.V., Suzuki, M.T., Hadd, A., Nguyen, L.P., Jovanovich, S.B., Gates, C.M., Feldman, R.A., Spudich, J.L., Spudich, E.N., DeLong, E.F.: Bacterial rhodopsin: evidence for a new type of phototrophy in the sea. Science 289, 1902–1906 (2000)
3. Turnbaugh, P.J., Hamady, M., Yatsunenko, T., Cantarel, B.L., Duncan, A., Ley, R.E., Sogin, M.L., Jones, W.J., Roe, B.A., Affourtit, J.P., Egholm, M., Henrissat, B., Heath, A.C., Knight, R., Gordon, J.I.: A core gut microbiome in obese and lean twins. Nature 457, 480–484 (2009)
4. Tatusov, R.L., Galperin, M.Y., Natale, D.A., Koonin, E.V.: The COG database: a tool for genome-scale analysis of protein functions and evolution. Nucleic acids research 28, 33–36 (2000)
5. Meyer, F., Paarmann, D., D'Souza, M., Olson, R., Glass, E.M., Kubal, M., Paczian, T., Rodriguez, A., Stevens, R., Wilke, A., Wilkening, J., Edwards, R.A.: The metagenomics RAST server - a public resource for the automatic phylogenetic and functional analysis of metagenomes. BMC bioinformatics 9, 386 (2008)
6. 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 research 36, D480–D484 (2008)
7. Kurokawa, K., Itoh, T., Kuwahara, T., Oshima, K., Toh, H., Toyoda, A., Takami, H., Morita, H., Sharma, V.K., Srivastava, T.P., Taylor, T.D., Noguchi, H., Mori, H., Ogura, Y., Ehrlich, D.S., Itoh, K., Takagi, T., Sakaki, Y., Hayashi, T., Hattori, M.: Comparative metagenomics revealed commonly enriched gene sets in human gut microbiomes. DNA Res. 14, 169–181 (2007)

8. Rodriguez-Brito, B., Rohwer, F., Edwards, R.A.: An application of statistics to comparative metagenomics. BMC bioinformatics 7, 162 (2006)
9. White, J.R., Nagarajan, N., Pop, M.: Statistical methods for detecting differentially abundant features in clinical metagenomic samples. PLoS computational biology 5, e1000352 (2009)
10. Gianoulis, T.A., Raes, J., Patel, P.V., Bjornson, R., Korbel, J.O., Letunic, I., Yamada, T., Paccanaro, A., Jensen, L.J., Snyder, M., Bork, P., Gerstein, M.B.: Quantifying environmental adaptation of metabolic pathways in metagenomics. Proc. Natl. Acad. Sci. USA 106, 1374–1379 (2009)
11. Tringe, S.G., von Mering, C., Kobayashi, A., Salamov, A.A., Chen, K., Chang, H.W., Podar, M., Short, J.M., Mathur, E.J., Detter, J.C., Bork, P., Hugenholtz, P., Rubin, E.M.: Comparative metagenomics of microbial communities. Science 308, 554–557 (2005)
12. Sharon, I., Pati, I., Markowitz, V.M., Pinter, R.Y.: A Statistical Framework for the Functional Analysis of Metagenomes. In: Batzoglou, S. (ed.) RECOMB 2009. LNCS, vol. 5541, pp. 496–511. Springer, Heidelberg (2009)
13. 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)
14. Dittrich, M.T., Klau, G.W., Rosenwald, A., Dandekar, T., Muller, T.: Identifying functional modules in protein-protein interaction networks: an integrated exact approach. Bioinformatics 24, i223–i231 (2008)
15. Storey, J.D., Tibshirani, R.: Statistical significance for genomewide studies. Proc. Natl. Acad. Sci. USA 100, 9440–9445 (2003)
16. Turnbaugh, P.J., Ley, R.E., Mahowald, M.A., Magrini, V., Mardis, E.R., Gordon, J.I.: An obesity-associated gut microbiome with increased capacity for energy harvest. Nature 444, 1027–1031 (2006)
17. Gallistl, S., Sudi, K., Mangge, H., Erwa, W., Borkenstein, M.: Insulin is an independent correlate of plasma homocysteine levels in obese children and adolescents. Diabetes Care 23, 1348–1352 (2000)
18. Eckel, R.H.: Obesity and heart disease: a statement for healthcare professionals from the Nutrition Committee, American Heart Association. Circulation 96, 3248–3250 (1997)
19. Borson-Chazot, F., Harthe, C., Teboul, F., Labrousse, F., Gaume, C., Guadagnino, L., Claustrat, B., Berthezene, F., Moulin, P.: Occurrence of hyperhomocysteinemia 1 year after gastroplasty for severe obesity. J. Clin. Endocrinol. Metab. 84, 541–545 (1999)
20. Mojtabai, R.: Body mass index and serum folate in childbearing age women. Eur. J. Epidemiol. 19, 1029–1036 (2004)
21. Tungtrongchitr, R., Pongpaew, P., Tongboonchoo, C., Vudhivai, N., Changbumrung, S., Tungtrongchitr, A., Phonrat, B., Viroonudomphol, D., Pooudong, S., Schelp, F.P.: Serum homocysteine, B12 and folic acid concentration in Thai overweight and obese subjects. Int. J. Vitam. Nutr. Res. 73, 8–14 (2003)
22. Hirsch, S., Poniachick, J., Avendano, M., Csendes, A., Burdiles, P., Smok, G., Diaz, J.C., de la Maza, M.P.: Serum folate and homocysteine levels in obese females with non-alcoholic fatty liver. Nutrition 21, 137–141 (2005)
23. Fokkema, M.R., Woltil, H.A., van Beusekom, C.M., Schaafsma, A., Dijck-Brouwer, D.A., Muskiet, F.A.: Plasma total homocysteine increases from day 20 to 40 in breastfed but not formula-fed low-birthweight infants. Acta Paediatr. 91, 507–511 (2002)

A Novel Approach for Compressing Phylogenetic Trees

Suzanne J. Matthews, Seung-Jin Sul, and Tiffani L. Williams

Texas A&M University, College Station TX 77843, USA
{sjm,sulsj,tlw}@cse.tamu.edu

Abstract. Phylogenetic trees are tree structures that depict relationships between organisms. Popular analysis techniques often produce large collections of candidate trees, which are expensive to store. We introduce TreeZip, a novel algorithm to compress phylogenetic trees based on their shared evolutionary relationships. We evaluate TreeZip's performance on fourteen tree collections ranging from 2,505 trees on 328 taxa to 150,000 trees on 525 taxa corresponding to 0.6 MB to 434 MB in storage. Our results show that TreeZip is very effective, typically compressing a tree file to less than 2% of its original size. When coupled with standard compression methods such as `7zip`, TreeZip can compress a file to less than 1% of its original size. Our results strongly suggest that TreeZip is very effective at compressing phylogenetic trees, which allows for easier exchange of data with colleagues around the world.

1 Introduction

Phylogenetics is concerned with reconstructing the evolutionary history (or family tree) for a set of organisms. An understanding of evolutionary mechanisms and relationships is at the heart of modern pharmaceutical research for drug discovery. It is also helping researchers understand (and defend against) rapidly mutating viruses such as HIV, and is the basis of genetically enhanced organisms. Typically, the evolutionary history for these organisms (or taxa) is depicted as a binary tree, where the taxa are the leaves of the tree and the edges represent the evolutionary relationships between the taxa (see Figure 1). To reconstruct a phylogenetic tree, the most popular techniques (such as MrBayes [5]) often return tens to hundreds of thousands of trees that represent equally-plausible hypotheses for how the taxa evolved from a common ancestor. We develop a new compression algorithm called *TreeZip* that reduces the requirements over standard compression algorithms for storing large collections of evolutionary trees. Furthermore, our TreeZip algorithm allows large phylogenetic tree collections to be shared easily with colleagues around the world.

The set of all edges (or *bipartitions*) from an evolutionary tree uniquely defines that tree. However, a tree's non-trivial bipartitions (or internal edges) are of most interest. To simplify our discussion, we use the term bipartitions to refer to a tree's set of non-trivial bipartitions. In Figure 1, each tree's bipartitions are represented by vertical lines. A bipartition represents a split on an internal

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 113–124, 2010.

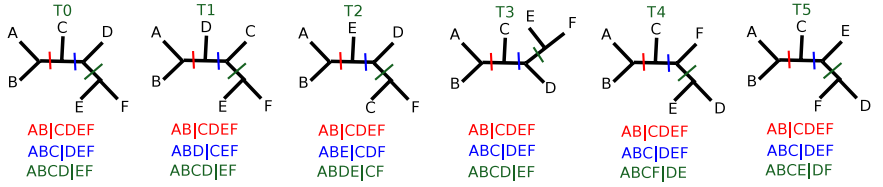


Fig. 1. A collection of six evolutionary trees on six taxa labeled A to F. For each tree, its set of bipartitions are listed.

T_0 = (((A,B),C),(D,(E,F)));
T_1 = (((A,B),D),(C,(E,F)));
T_2 = (((A,B),E),(D,(C,F)));
T_3 = (((A,B),C),((E,F),D));
T_4 = (((A,B),C),(F,(E,D)));
T_5 = (((A,B),C),(E,(F,D)));

(a) Newick strings

T_0 = (D,((A,B),C),(E,F));
T_1 = (C,(D,(B,A)),(E,F));
T_2 = (D,((B,A),E),(F,C));
T_3 = (D,(C,(B,A)),(E,F));
T_4 = (F,((A,B),C),(E,D));
T_5 = (E,((B,A),C),(D,F));

(b) equivalent Newick strings

Fig. 2. Newick representations for the phylogenetic trees shown in Figure 1. Two different, but equivalent, Newick representations are given for each tree.

edge of the evolutionary tree that separates the taxa into two groups. A set of bipartitions uniquely defines an evolutionary tree. For example, tree T_0's bipartitions are $AB|CDEF$, $ABC|DEF$, and $ABCD|EF$ where the symbol '|' acts as a separator. Trees T_0 and T_3 are identical trees since they contain the same set of bipartitions. For a binary tree, the number of bipartitions is $n-3$, where n is the number of taxa.

The Newick format [4] is the most widely used format to store a phylogenetic tree in a file. In this format, the topology of the evolutionary tree is represented using a notation based on balanced parentheses. Consider tree T_0 in Figure 1. A Newick representation of the topology of this tree is `(((A,B),C),(D,(E,F)));`, where '`;`' symbolizes the end of the Newick string. Matching pairs of parentheses symbolize internal nodes in the evolutionary tree. The Newick representation of a tree is not unique. For example, another valid Newick string for tree T_0 is `(D,((A,B),C),(E,F));`. Figure 2(a) shows the Newick tree file for the trees in Figure 1, where the Newick representation is based on the lexicographical ordering of the taxa names. Given that trees can have multiple, valid Newick strings, Figure 2(b) shows a different Newick file, where the taxa names are ordered randomly for each tree. For a given tree T_i on n taxa, there are $O(2^{n-1})$ possible Newick strings to represent it.

Our contributions. In this paper, we introduce TreeZip, a new lossless algorithm for compressing large collections of phylogenetic trees. TreeZip requires $O(nt)$ running time for both its compression and decompression phases, where n is the number of taxa and t is the number of trees in the collection of interest.

Given that many of the bipartitions in a collection of phylogenetic trees are shared, the *novelty* of our TreeZip approach is storing such relationships only once in the compressed representation. TreeZip compresses a Newick file based on the *semantic* representation (i.e., tree bipartitions). General-purpose data compression techniques (e.g., `gzip`, `bzip,` and `7zip`) do not know what the data (Newick file) represents beyond the ASCII string representations. Hence, there is great potential for obtaining good compression by utilizing the semantic information in a Newick file describing large collections of evolutionary trees.

TreeZip leverages two phylogenetic tree algorithms, HashCS (constructs consensus trees) [11] and HashRF (computes a topological $t \times t$ distance matrix) [10], which use a hash table to organize the bipartitions from a collection of trees efficiently. We demonstrate the performance of our TreeZip algorithm in comparison to standard compression approaches (i.e., `gzip`, `bzip,` and `7zip`) on 14 different large-scale tree collections. Our largest (smallest) tree collection consists of 150,000 (2,505) trees requiring 434 MB (0.6 MB) of storage space. Overall, our results show that the compressed TreeZip (`.trz`) file occupies from 0.2% to 2% of its original size, which outperforms `gzip` and `bzip` compression algorithms. When TreeZip is coupled with a standard compression algorithm, even greater compression is attained. For the datasets studied here, the best compression occurs when TreeZip compression is followed by `7zip`. Hence, TreeZip is a great alternative for biologists who want to recycle the trees generated from their experiments.

Related work. To the best of our knowledge, the Texas Analysis of Symbolic Phylogenetic Information (TASPI) [2] [3] is the only described approached for compressing evolutionary trees. It is written in the ACL2 formal logic language, but we were unable to find an available implementation of the TASPI algorithm for direct comparison to our approach on all of our tree collections. However, we were able to obtain the collection of trees that TASPI used to evaluate their approach [2]. Section 3.1 compares the compression ratios of TreeZip to TASPI on those set of trees, but without a TASPI implementation, we were unable to compare running times.

Our TreeZip algorithm compliments and extends the work done with TASPI in several ways. While compression storage results are given, the main focus of TASPI is building a single consensus (or summary) tree from a compressed representation of the collection of trees. While TreeZip can build consensus trees (not shown here), our main focus is on compressing large collections of evolutionary trees efficiently. Since a Newick string does not give a unique representation for a phylogenetic tree (there are $O(2^{n-1})$ possible Newick strings), the designers of TASPI note that their algorithm is affected by the ordering of the taxa in the Newick string. TreeZip, on the other hand, has been designed to not be impacted by different Newick strings representing the same tree. Finally, TASPI does not explicitly state if it has a decompression routine in order to rebuild the original Newick tree file containing the t trees. TreeZip has such a routine.

2 Our TreeZip Algorithm

Our TreeZip algorithm compresses and decompresses phylogenetic trees based on their shared evolutionary relationships. Under compression, the input to the algorithm is a Newick file and the output is a TreeZip (or a `.trz`) file. The input to TreeZip's decompression phase is a `.trz` file and the output is a Newick file.

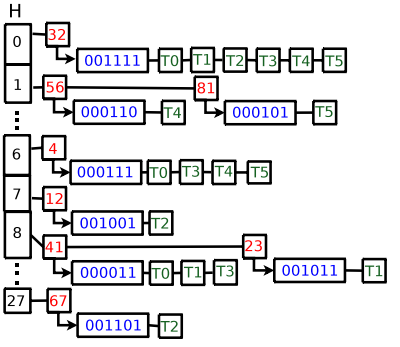
2.1 Compression: Converting the Newick File to a `.trz` File

Building a hash table from the Newick file. In the Newick input file, each string i, which represents tree T_i, is read and stored in a tree data structure. During the depth-first traversal of input tree T_i, each of its bipartitions is fed through two universal hash functions, h_1 and h_2 [1]. Both of these functions require as input a n-bit bitstring representation of each bipartition in tree T_i, where n represents the number of taxa. In the n-bit bitstring, the first bit is labeled by the first taxon name, the second bit is represented by the second taxon, etc. We can represent all of the taxa on one side of the tree with the bit '0' and the remaining taxa on the side of the tree with the bit '1'. In our example, taxa on the same side of a bipartition as taxon A receive a '0'. In Figure 1 tree T_1's bipartitions are $AB|CDEF$, $ABD|CEF$, and $ABCD|EF$ which can be described by the bitstrings 001111, 001011, and 000011, respectively.

The hash function h_1 is used to generate the location (index) for storing a bipartition in the hash table. h_2 is responsible for creating a unique and short bipartition identifier (BID) for the bipartition so that the entire n-bit bitstring does not have to be analyzed in order to insert bipartitions into the hash table. Our two universal hash functions are defined as follows: $h_1(B) = \sum b_i a_i \bmod m_1$ and $h_2(B) = \sum b_i a_i \bmod m_2$, where $A = (a_1, ..., a_n)$ is a list of random integers in $(0, ..., m_1\text{-}1)$ and $B = (b_1, ..., b_n)$ is a bipartition represented as an n-bit bitstring. m_1 represents the number of entries (or locations) in the hash table. m_2 represents the largest bipartition ID (BID) given to a bipartition. b_i represents the ith bit of the n-bit bitstring representation of the bipartition B.

Figure 3(a) shows how the bipartitions from Figure 1 are stored in our hash table. Each entry in the hash table consists of BID, a bitstring representation of the bipartition, and a list of trees that contain that bipartition. Using these universal hash functions, the probability that any two distinct bipartitions B_i and B_j collide (i.e., $h_1(B_i) = h_1(B_j)$) is $\frac{1}{m_1}$. In Figure 3, $H[1]$ and $H[8]$ show two different bipartitions colliding to the same location in the hash table. Bipartitions $ABCF|DE$ and $ABCE|DF$ both reside in $H[1]$ and $ABCD|EF$ and $ABD|CEF$ reside in $H[8]$. However, these colliding bipartitions are differentiated by their h_2 hash value. In location $H[1]$, h_2 values 56 and 81 differentiate bipartitions $ABCF|DE$ and $ABCE|DF$, respectively.

The probability of a double collision ($h_1(B_i) = h_1(B_j)$ and $h_2(B_i) = h_2(B_j)$) is $O(\frac{1}{c})$, where c can be an arbitrarily large number [1]. Double collisions often result in an incorrect result for the underlying application. In our experience with using these hash functions in our phylogenetic applications (HashCS, HashRF,



(a) Hash table

1. TAXA␣A:B:C:D:E:F
2. NTREES␣6
3. NBIPARTITIONS␣8
4. L2K4␣-:0:
5. L3K2A␣+:1:E
6. L3BAB␣+:1:F
7. L3K3␣-:2:K2
8. L2BL2B␣+:1:C
9. L4K2␣+:3:ABC
10. L2BAK2␣+:1:B
11. L2K2AB␣+:1:C

(b) TreeZip (`.trz`) file

Fig. 3. TreeZip compressed file, which was obtained from our hash table data structure, for the phylogenetic trees shown in Figure 1. The symbol ␣ represents a visible space that is in the TreeZip file.

and TreeZip), we have not encountered double collision even when using small c values. For t trees on n taxa, $O(nt)$ time is required to construct the hash table.

Converting the hash table to `.trz` *format.* Once all of the bipartitions are organized in the hash table, we can begin the process of writing the `.trz` compressed file, which is binary. Figure 3(b) shows a compressed version of the hash table in Figure 3(a). The first three lines of the `.trz` file represent the taxa names, the number of trees in the file, and the number of unique bipartitions denoted by lines 1–3 in the `.trz` file in Figure 3(b). The remaining lines in the `.trz` file are related to the bipartitions contained in the t evolutionary trees. Each of the remaining lines is composed of two parts (n-bit bitstring and list of tree ids) separated by a single space.

We run-length encode our bitstrings. Run-length encoding is a form of data compression in which runs of data (i.e, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run. For the bitstring 001111 in Figure 3, we would have a run-length encoding of 0:2 1:4, where each $x : y$ element represents the data value (x) and the number of repetitions (y). Since bitstrings can either contain runs of 1s or 0s, we introduce two new symbols. `1:` is encoded as K, while `0:` encoded as L. (We use characters A through J for compressing our list of tree ids described shortly.) Hence, we encode the bitstring 001111 as L2K4. In our experiments, we considered taking every group of 7 bits in our bitstring and translating it to an ASCII character. However, we were able to get better compression by using run-length encoding, which showed significant benefits on our biological tree collections consisting of thousands of taxa.

The set of unique bipartitions comprise the remaining portion of the `.trz` file. Let $\mathcal{T}$ represent the set of evolutionary trees of interest, where $|\mathcal{T}| = t$. For a bipartition B, $\mathcal{B}_{in}$ represents the set of the trees in $\mathcal{T}$ that share that bipartition. $\mathcal{B}_{out}$ is the set of trees that do not share bipartition B. Since these

sets are complements, their union comprises the set $\mathcal{T}$. To minimize the amount of information present in our `.trz` output, we print out the contents of the smaller of these two sets. If $|\mathcal{B}_{in}| \leq |\mathcal{B}_{out}|$, then we output $\mathcal{B}_{in}$. Otherwise, $\mathcal{B}_{out}$ is outputted. In our `.trz` file, we denote $\mathcal{B}_{in}$ and $\mathcal{B}_{out}$ lines with the `'+'`and `'-'` symbol, respectively.

Even with use of the smaller of the $\mathcal{B}_{in}$ or $\mathcal{B}_{out}$ sets, the list of tree ids can get very large. This is due to the fact that as t grows large, the number of bytes necessary to store a single id also grows. Since the trees are inserted into the hash table in their order of appearance in the Newick file, our lists of tree ids will be in increasing order. As a result, we store the differences between adjacent elements in our tree id list. These differences are then run-length encoded. To eliminate the need for spaces between the run-length encoded differences, the first digit of every element is encoded as a character, with $0 \ldots 9$ represented by $\mathrm{A} \ldots \mathrm{J}$. Consider bipartition $ABCD|EF$ (bitstring 000011), which is in $H[8]$ in Figure 3. The $\mathcal{B}_{in}$ set will be used for this bipartition, and its run-length encoded differences will be 0 1 2, which will be encoded as `ABC` on line 9 in the `.trz` file.

Finally, one of the guiding factors for our TreeZip format is not only effective compression, but also readability. We did try several different compression schemes for our TreeZip approach, but the compression algorithm described here gave the best compression along with the best decompression times (not shown).

2.2 Decompression: Converting the `.trz` File to a Newick File

Two major steps of the decompression in TreeZip are decoding the contents in the `.trz` file and rebuilding the collection of t trees. Decoding reconstructs the original hash table information which consists of bitstrings and the tree ids that contain them. When the `.trz` file is decoded, each line of the file is processed sequentially. First, the taxa information is fed into TreeZip. Next, the number of trees is read. Each bipartition is then read sequentially.

To assist in bipartition collection, we maintain two data structures. The first, which we will refer to as V, is a vector of the bipartitions contained in *all* of the t trees. The second, M, is a $t \times k$ matrix, where $k = n - 3$ is the maximum possible number of bipartitions for a phylogenetic tree. The length of the matrix M corresponds to the number of trees specified in the `.trz` file. Each row i in matrix M corresponds to the bipartitions required to rebuild tree T_i. For example, in figure 3, the bipartition 000011 is shared among all the trees. It is therefore added to vector V. On the other hand, the bipartitions on lines 5 and 6 are contained in only trees 4 and 5 respectively, and therefore will be added to $M[4]$ and $M[5]$. The bipartition on line 9 will be added to $M[0], M[1]$, and $M[3]$ since `ABC` decodes to the tree ids T_0, T_1, and T_3. Line 7 in our `.trz` file warrants special attention. Since this line belongs to the set $\mathcal{B}_{out}$, we know upon decoding that this bipartition does *not* belong to trees 1 and 2. Therefore, the bipartition is added to rows $M[0], M[3], M[4]$, and $M[5]$.

The decoded bitstrings are the basic units for building trees. Once the bitstrings and the associated tree ids are decoded, we can build the original trees one by one. In order to build tree x, the tree building function receives as input

the vector V containing bipartitions shared among all of the trees and matrix row $M[x]$ which contains the bipartitions encoded as bitstrings for tree x. Since vector V contains the bitstrings common to all the trees, it is always passed to the tree building function.

Each of the t trees is built starting from tree T_0 and ending with tree T_{t-1}, whose bipartitions are stored in $M[0]$ and $M[t-1]$, respectively. The trees are reconstructed in the same order that they were in the original Newick file. However, given $O(2^{n-1})$ possible Newick strings for a tree T_i, the Newick representation that TreeZip outputs for tree T_i will probably differ from the Newick string in the original file. However, this is not a problem semantically since the different strings represent the same tree.

In order to build tree T_i, the bitstrings in matrix $M[i]$ and vector V are merged into a single array of bitstrings. Initially, tree T_i is represented as a star tree on n taxa. Bipartitions from $M[i]$ are added to refine tree T_i based on the number of 1's in its bitstring representation. (The number of 0's could have been used as well.) The more 1's in the bitstring representation, the more taxa that are grouped together by this bipartition. A star tree is an bitstring representation consisting of all 1's. For each of T_i's bitstrings, we count the number of 1's it contains. Bipartitions are then sorted in increasing order of their bitstrings, which means that bipartitions that group together the most taxa appear first. The bipartition that groups together the fewest taxa appears last in the sorted list of '1' bit counts. For each bipartition, a new internal node in tree T_i is created. Hence, the bipartition is scanned to put the taxa into two groups—taxa with '0' bits compose one group and those with '1' bits compose the other group. The taxa indicated by the '1' bits become children of the new internal node. The above process repeats until all bipartitions are added to tree T_i.

3 Experimental Results

Our implementation of TreeZip used in the following experiments can be found at `http://treezip.googlecode.com`. Experiments were conducted on a 2.5Ghz Intel Core 2 quad-core machine with 4GB of RAM running Ubuntu Linux 8.10. We ran our experiments on fourteen sets of trees which are described in Table 1. We use the *compression ratio* measure to evaluate the performance of TreeZip in comparison to general-purpose compression algorithms. The compression ratio C is calculated as $C = \frac{|\text{compressed file}|}{|\text{original file}|}$. This result is multiplied by 100 to achieve a percentage. A lower compression ratio denotes better compression of the original file.

3.1 Performance on the TASPI Tree Collection

In Figure 4, we compare the compression ratio achieved by TreeZip and TASPI on the 9 tree collections used in Collection 3 of [2], which is denoted by datasets 6 through 14 in Table 1. We also show the compression ratio of standard compression approaches (`gzip`, `bzip,` and `7zip`) achieved on this set of trees, along with

Table 1. Characteristics of our biological tree files. The `mammals`, `freshwater`, `angiosperms`, `fish`, and `insects` datasets were given to us by biologists. The remaining tree collections are the same ones used by Boyer et al. to evaluate their TASPI approach.

	Datasets	Description	Taxa	Trees	File size (MB)	Bipartitions
1	`mammals`	Mammalian trees [6]	16	8,000	0.6	13
2	`freshwater`	Organisms from freshwater, marine, and oil habitats [7]	150	20,000	16.0	1,168
3	`angiosperms`	Flowering plants [9]	567	33,306	105.0	2,444
4	`fish`	Fish trees (unpublished collection from M. Glasner's lab at Texas A&M)	264	90,000	127.0	12,115
5	`insects`	Insect trees [8]	525	150,000	434.0	574
6	`aster328`	Tree Collection 3 from Boyer et al. [2]	328	2,505	5.3	788
7	`eern476`		476	2,505	7.7	3,019
8	`john921`		921	2,505	16.0	15,448
9	`lipsc439`		439	2,505	7.1	903
10	`mari2594`		2,594	2,505	47.0	8,628
11	`ocho854`		854	2,505	15.0	3,232
12	`rbcl500`		500	2,505	8.2 (8.1 in [2])	1,579
13	`three567`		567	2,505	9.3	1,588
14	`will2000`		2,000	2,505	36.0	13,257

the ratio of TreeZip coupled with each of these standard approaches. Since an implementation of TASPI is not available publicly, the compression ratio numbers for TASPI were calculated directly from [2]. Since TASPI coupled its approach with the `bzip` algorithm, we highlight the compression ratio achieved by TASPI and TASPI+`bzip` (blue), as well as TreeZip and TreeZip+`bzip` (red). TASPI did not couple its approach with either `7zip` or `gzip`.

TreeZip achieves a better (lower) compression ratio than TASPI across all the listed datasets. For example, on the `lipsc439` dataset, TreeZip achieves a compression ratio of 1.592%, while TASPI achieves a compression ratio of 5.57%. This corresponds to a file size of 116 kilobytes and 406 kilobytes respectively. On the `mari2594` dataset, TreeZip achieves a compression ratio of 2.34%, compared to TASPI's 7.02%. This corresponds to compressed file sizes of 1.1 MB and 3.3 MB respectively.

When coupled with `bzip`, TASPI achieves a slightly better compression ratio than TreeZip+`bzip` on most of the datasets. However, these differences are often negligible. For example, on the `three567` dataset, TreeZip+`bzip` has a compression ratio of 0.63% compared to TASPI+`bzip`'s 0.47%. This corresponds to 60 and 45 kilobytes respectively, a difference of 15 kilobytes. On the `lipsc439` dataset, TreeZip+`bzip` achieves a compression ratio of 0.55%, compared to TASPI+`bzip`'s 0.48%. This corresponds to compressed files of 40 and

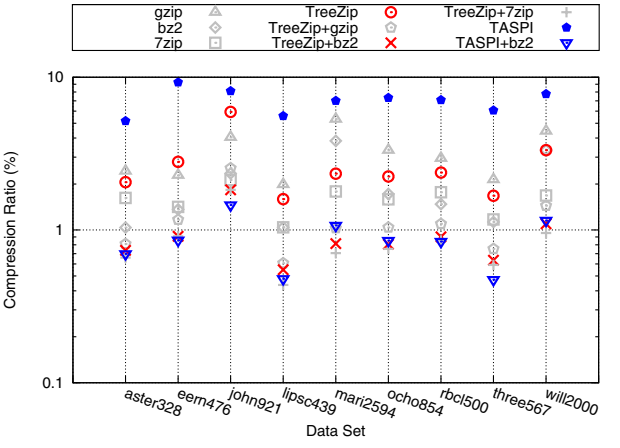


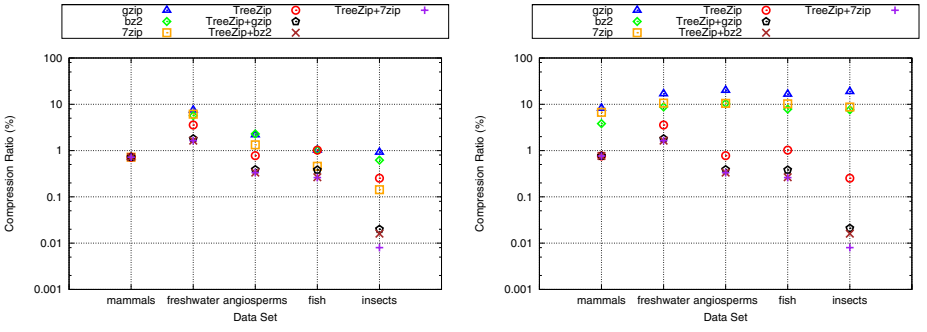
Fig. 4. Compression ratios for various algorithms on Newick string representations of evolutionary trees. TASPI and TASPI+bz2 numbers come from [2].

34.8 kilobytes in size, respectively. On the largest dataset of this set, `mari2594`, TreeZip+`bzip` outperforms TASPI, achieving a compression ratio of 0.81% compared to TASPI+`bzip`'s 1.07%. This corresponds to file sizes of and 392 and 515 kilobytes, respectively, a difference of 123 kilobytes.

3.2 Performance on Tree Sets Provided by Biologists

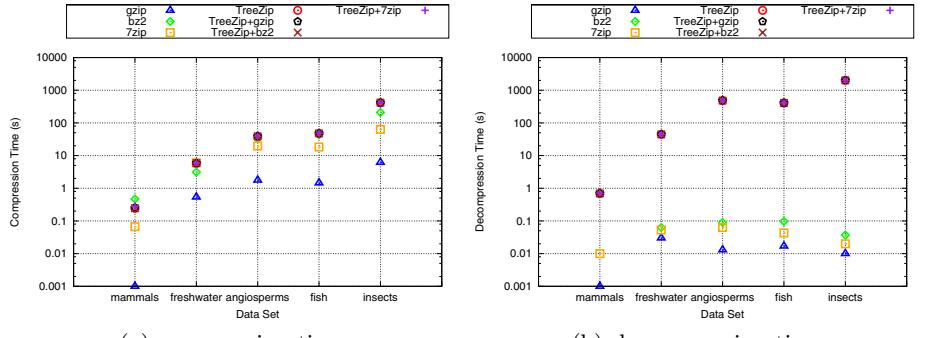
Figure 5(a) shows the performance of TreeZip on the large tree collections (Datasets 1 through 5 in Table 1) given to us by biologists. By itself, TreeZip achieves similar storage to the standard compression algorithms on our biological tree sets. Since all of the trees in the `mammals` dataset are identical, all approaches achieve the same compression ratio and storage size of 4 kilobytes. For our `fish` dataset, `7zip` outperformed TreeZip and the other standard compression approaches, achieving a ratio of 0.46%. TreeZip, on the other hand, had a compression ratio of 1.02%. This corresponds to a size of 596 kilobytes compared to TreeZip's 1.3 megabytes. Coupling TreeZip with standard compression techniques results in improved performance. Returning back to our `fish` dataset, TreeZip+`7zip` achieves a compression ratio of 0.261%, which corresponds to 340 kilobytes. This is most evident for our `insects` dataset, where TreeZip+`7zip` achieves a compression ratio of 0.008%, or roughly 36 kilobytes. On this same dataset, `7zip` has a compression ratio of 0.14% resulting in a compressed file of 636 kilobytes. Our results suggest that the greater the level of bipartition sharing and the number of trees, the better TreeZip will perform, especially when coupled with the `7zip` approach.

One critical advantage of TreeZip is that it collapses the topologies of the phylogenetic trees into a set of common bipartitions, ensuring that each bipartition appears at most once in the compressed form. Both standard compression techniques and TASPI compress trees at the *string* level. If the Newick string for



(a) original Newick file (b) different, but equivalent, Newick file

Fig. 5. Compression ratios of two different Newick files representing the same set of evolutionary trees



(a) compression time (b) decompression time

Fig. 6. Compression and decompression times for the algorithms under study

a particular tree is rearranged denoting a different, but equivalent, Newick string representation of the same tree, text-based compression approaches will have difficulty identifying shared bipartitions among the t trees. Figure 5(b) shows the impact of using different, but equivalent Newick representations (see Figure 2) in our biological tree collections. While TreeZip's performance remains the same, the compression ratio and storage requirements for the standard compression methods explode. For example, for the `fish` dataset, `7zip`'s compression ratio increases from 0.46% to 10.24%. This corresponds to an increase from 596 kilobytes to 13 megabytes. TASPI's storage requirements would also increase under different, but equivalent, Newick strings. In contrast, TreeZip and TreeZip+`7zip` still requires only 1.3 megabytes and 340 kilobytes of storage, respectively.

While TreeZip competes against standard compression algorithms in terms of storage size, it does so at the cost of running time (see Figure 6). While TreeZip's compression speed is about twice as slow as `bzip` and `7zip` (`gzip` runs extremely fast requiring less than 10 seconds on our datasets), its decompression speed is

very slow. All of the methods require less than a second to decompress, while TreeZip can take anywhere from a second to 3,000 seconds. Obviously, this is a place that needs optimization. However, if the user is merely interested in compressing tree files as part of an archive with very little chance for decompression, then TreeZip is a desirable alternative to standard compression techniques. Furthermore, if there is no predefined ordering of the taxa in the Newick file, then using TreeZip will result in a very small file compared to the alternatives given the robustness of the TreeZip approach.

4 Conclusions and Future Work

Phylogenetic heuristics often produces tens to hundreds of thousands of equally-plausible trees, which are usually stored in a Newick-formatted text file. Due to the number of trees, the size of the input file is easily over hundreds of megabytes making it difficult to store, maintain, and exchange the tree files. In this paper, we introduce our *TreeZip* algorithm, a novel approach that leverages the semantic information among trees to compress the tree files. The advantage of TreeZip over current methods is its ability to uniquely identify shared bipartitions and store this information in a compressed TreeZip (`.trz`) file, which consumes considerably less storage space than the original Newick file.

Our TreeZip algorithm outperforms standard compression methods by achieving a better compression ratio. For example, our results show that our `.trz` file occupies from 0.2% to 2% of the original Newick file, which outperforms `gzip` and `bzip` algorithms. When TreeZip is coupled with standard compression algorithms, it is the best compression technique for phylogenetic trees. Thus, TreeZip can work on two levels. It can work at the `.trz` file level, where the file can be used as input for other phylogenetic tree algorithms. The benefit of the `.trz` file is that it is readable and can be queried more easily (without decompression) than the Newick file regarding the evolutionary relationships contained in the collection of t trees. Coupling TreeZip with text compression algorithms such as `7zip` produces the best storage savings. In addition, a phylogenetic tree can be represented using several different (yet equivalent) Newick string representations. This proves disastrous for standard compression methods, which perform poorly in the absence of any available redundancy at the Newick string level. TreeZip, on the other hand, performs well on such datasets.

Overall, TreeZip's efficient method for compressing trees allows large phylogenetic tree collections to be easily exchanged with others, an essential component for successful scientific collaborations. Without compression, sharing data can become quite tedious, especially across long distances. As biologists obtain more data and use phylogenetic heuristics to build large-scale evolutionary trees, the size of their tree collections will continue to grow in size. Thus, compression algorithms such as TreeZip will become critical tools for helping biologists manage their rapidly expanding phylogenetic tree collections.

In the future, we will optimize TreeZip for speed since the focus in this work was the quality of the compression achieved. Due to the inherent flexibility of

the compressed format, we plan to add in functionality to incorporate new, additional trees into an existing compressed collection.

Acknowledgments

Funding for this project was supported by NSF under grants DEB-0629849 and IIS-0713618.

References

1. Amenta, N., Clarke, F., John, K.S.: A linear-time majority tree algorithm. In: Benson, G., Page, R.D.M. (eds.) WABI 2003. LNCS (LNBI), vol. 2812, pp. 216–227. Springer, Heidelberg (2003)
2. Boyer, R.S., Hunt Jr., W.A., Nelesen, S.: A compressed format for collections of phylogenetic trees and improved consensus performance. Technical Report TR-05-12, Department of Computer Sciences, The University of Texas at Austin (2005)
3. Boyer, R.S., Hunt Jr., W.A., Nelesen, S.: A compressed format for collections of phylogenetic trees and improved consensus performance. In: Casadio, R., Myers, G. (eds.) WABI 2005. LNCS (LNBI), vol. 3692, pp. 353–364. Springer, Heidelberg (2005)
4. Felsenstein, J.: The Newick tree format. Internet Website (last accessed January 2010), Newick, http://evolution.genetics.washington.edu/phylip/newicktree.html
5. Huelsenbeck, J.P., Ronquist, F.: MRBAYES: Bayesian inference of phylogenetic trees. Bioinformatics 17(8), 754–755 (2001)
6. Janecka, J.E., Miller, W., Pringle, T.H., Wiens, F., Zitzmann, A., Helgen, K.M., Springer, M.S., Murphy, W.J.: Molecular and genomic data identify the closest living relative of primates. Science 318, 792–794 (2007)
7. Lewis, L.A., Lewis, P.O.: Unearthing the molecular phylodiversity of desert soil green algae (chlorophyta). Syst. Bio. 54(6), 936–947 (2005)
8. Molin, A.D., Matthews, S., Sul, S.-J., Munro, J., Woolley, J.B., Heraty, J.M., Williams, T.L.: Large data sets, large sets of trees, and how many brains? – Visualization and comparison of phylogenetic hypotheses inferred from rdna in chalcidoidea (hymenoptera) (poster December 2009), http://esa.confex.com/esa/2009/webprogram/Session11584.html
9. Soltis, D.E., Gitzendanner, M.A., Soltis, P.S.: A 567-taxon data set for angiosperms: The challenges posed by bayesian analyses of large data sets. Int. J. Plant Sci. 168(2), 137–157 (2007)
10. 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)
11. Sul, S.-J., Williams, T.L.: An experimental analysis of consensus tree algorithms for large-scale tree collections. In: Măndoiu, I., Narasimhan, G., Zhang, Y. (eds.) ISBRA 2009. LNCS, vol. 5542, pp. 100–111. Springer, Heidelberg (2009)

Structure of Proximal and Distant Regulatory Elements in the Human Genome
(Invited Keynote Talk)

Ivan Ovcharenko

Computational Biology Branch, National Center for Biotechnology Information
National Library of Medicine, National Institutes of Health
8600 Rockville Pike, Bethesda, MD 20896
ovcharen@nih.gov

Clustering of multiple transcription factor binding sites (TFBSs) for the same transcription factor (TF) is a common feature of cis-regulatory modules in invertebrate animals, but the occurrence of such homotypic clusters of TFBSs (HCTs) in the human genome has remained largely unknown. To explore whether HCTs are also common in human and other vertebrates, we used known binding motifs for vertebrate TFs and a hidden Markov model-based approach to detect HCTs in the human, mouse, chicken, and fugu genomes, and examined their association with cis-regulatory modules. We found that evolutionarily conserved HCTs occupy nearly 2% of the human genome, with experimental evidence for individual TFs supporting their binding to predicted HCTs. More than half of promoters of human genes contain HCTs, with a distribution around the transcription start site in agreement with the experimental data from the ENCODE project. In addition, almost half of 487 experimentally validated developmental enhancers contain them as well - a number more than 25-fold larger than expected by chance. We also found evidence of negative selection acting on TFBSs within HCTs, as the conservation of TFBSs is stronger than the conservation of sequences separating them. The important role of HCTs as components of developmental enhancers is additionally supported by a strong correlation between HCTs and the binding of the enhancer-associated co-activator protein p300. Experimental validation of HCT-containing elements in both zebrafish and mouse suggest that HCTs could be used to predict both the presence of enhancers and their tissue specificity, and are thus a feature that can be effectively used in deciphering the gene regulatory code. In conclusion, our results indicate that HCTs are a pervasive feature of human cis-regulatory modules and suggest that they play an important role in gene regulation in the human and other vertebrate genomes.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, p. 125, 2010.

Combinatorics in Recombinational Population Genomics

(Invited Keynote Talk)

Laxmi Parida

Computational Genomics
IBM T.J. Watson Research, Yorktown Heights, USA
parida@us.ibm.com

The work that I will discuss is motivated by the need for understanding, and processing, the manifestations of recombination events in chromosome sequences. In this talk, we focus on two related problems. First, we explore the very general problem of reconstructability of pedigree history. How plausible is it to unravel the history of a complete unit (chromosome) of inheritance? The second problem deals with reconstructing the recombinational history of a collection of chromosomes.

For the first problem, we use a random graphs framework to study pedigree history in an ideal (Wright Fisher) population [Par09]. This framework correlates the underlying mathematical objects in pedigree graph, mtDNA or NRY Chr tree, ARG (Ancestral Recombinations Graph), HUD etc. used in literature, into a single unified random graph framework. It also gives a natural definition, based solely on the topology, of an ARG, one of the most interesting as well as useful mathematical object in this area [GM97]. The random graphs framework gives an alternative parametrization of the ARG that does not use the recombination rate ρ and instead uses a parameter M based on the (estimate of) the number of non-mixing segments in the extant units [GSN+02]. These non-mixing segments may also be viewed as identity by descent (IBD) segments. This framework also gives a purely topological definition of GMRCA, analogous to MRCA on trees (which has a purely topological description i.e., it is a root, graph-theoretically speaking, of a tree). An interesting fallout of this study is a randomized algorithm leading to a combinatorial construction of the ARG. I will describe a purely combinatorial construction of the ARG based on the coalescent approach [PJ10]. An important departure from the earlier models [Hud90, HSW05] is the use of the integer parameter M instead of the recombination rate parameter ρ. The appeal of the classical coalescence theory is many. From a simulation perspective, it is the elegant elicitation of population dynamics, without any explicit biology. In coalescence-based simulation, mutations and other genetic events are decorated on the random combinatorial (tree) structure. Recombination rate(s), however is a parameter based on the biology of the organisms in the population. In an effort to isolate detailed biology from the population dynamics, this approach constructs a random combinatorial (DAG) structure and then decorates it with the biological events.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 126–127, 2010.

The second problem addresses the following question. As large databases become available, is it possible to reconstruct the pedigree history in the data [PMC+08, PJM+09]. And, what stories, if any, these recombinational landscapes tell us [MJC+09]. We exploit the coherence that is observed in the human haplotypes as patterns and present a network model of patterns to reconstruct the ARG. I will conclude with a discussion on our ongoing work in the Genographic Project on the study of human population diversity based on evidence of past recombinations (termed recotypes) as genetic markers. The inferred recombinations indicate strong agreement with past in vitro and in silico recombination rate estimates. The correlation between traditional allele frequency based distances and recombinational distances bring further credence to the study of population structure using recotypes. Also, we make the surprising observation that recotypes are more representative of the underlying population structure than the haplotypes they are derived from.

References

[GM97] Griffiths, R.C., Marjoram, P.: An ancestral recombinations graph. In: Donnelly, P., Tavare, S. (eds.) Progress in Population Genetics and Human Evolution, IMA vols in Mathematics and its Applications, vol. 87, pp. 257–270 (1997)

[GSN+02] Gabriel, S.B., Schaffner, S.F., Nguyen, H., Moore, J.M., Roy, J., Blumenstiel, B., Higgins, J., De Felice, M., Lochner, A., Faggart, M., Liu-Cordero, S.N., Rotimi, C., Adeyemo, A., Cooper, R., Ward, R., Lander, E.S., Daly, M.J., Altshuler, D.: The structure of haplotype blocks in the human genome. Science 296(5576), 2225–2229 (2002)

[Hud90] Hudson, R.R.: Gene genealogies and the coalescent process. In: Oxford Surveys in Evolutionary Biology. Oxford University Press, Oxford (1990)

[HSW05] Hein, J., Schierup, M.H., Wiuf, C.: Gene Genealogies, Variation and Evolution: A Primer in Coalescent Theory. Oxford Press, Oxford (2005)

[MJC+09] Melé, M., Javed, A., Calafell, F., Parida, L., Bertranpetit, J.: Genographic Consortium. Recombination-based genomics: a genetic variation analysis in human populations (2009) (under submission)

[Par09] Parida, L.: Ancestral Recombinations Graph: A Reconstructability Perspective using Random-Graphs Framework (2009) (under submission)

[PJ10] Parida, L., Javed, A.: Coalescence with Recombinations: Combinatorial Construction of ARGs (2010) (under submission)

[PMC+08] Parida, L., Melé, M., Calafell, F., Bertranpetit, J.: Genographic Consortium. Estimating the Ancestral Recombinations Graph (ARG) as Compatible Networks of SNP Patterns. Journal of Computational Biology 15(9), 1–22 (2008)

[PJM+09] Parida, L., Melé, M., Calafell, F., Bertranpetit, J.: Genographic Consortium. Minimizing recombinations in consensus networks for phylogeographic studies. BMC Bioinformatics 10(1), S72 (2009) doi = 10.1186/1471-2105-10-S1-S72

Uncovering Hidden Phylogenetic Consensus

Nicholas D. Pattengale[1], Krister M. Swenson[2,3], and Bernard M.E. Moret[4]

[1] Department of Computer Science, University of New Mexico, USA
[2] Department of Mathematics and Statistics, University of Ottawa, Canada
[3] LaCIM, Université du Québec à Montréal, Canada
[4] Laboratory for Computational Biology and Bioinformatics, EPFL, Switzerland

Abstract. Many of the steps in phylogenetic reconstruction can be confounded by "rogue" taxa, taxa that cannot be placed with assurance anywhere within the tree—whose location within the tree, in fact, varies with almost any choice of algorithm or parameters. Phylogenetic consensus methods, in particular, are known to suffer from this problem. In this paper we provide a novel framework in which to define and identify rogue taxa. In this framework, we formulate a bicriterion optimization problem that models the net increase in useful information present in the consensus tree when certain taxa are removed from the input data. We also provide an effective greedy heuristic to identify a subset of rogue taxa and use it in a series of experiments, using both pathological examples described in the literature and a collection of large biological datasets. As the presence of rogue taxa in a set of bootstrap replicates can lead to deceivingly poor support values, we propose a procedure to recompute support values in light of the rogue taxa identified by our algorithm; applying this procedure to our biological datasets caused a large number of edges to change from "unsupported" to "supported" status, indicating that many existing phylogenies should be recomputed and reevaluated to reduce any inaccuracies introduced by rogue taxa.

1 Introduction

Phylogenetic consensus methods are used for combining a set of trees defined on the same set of leaves into a single tree that summarizes the information found in the set. By their very nature, these methods discard information, typically structural elements not prevalent in the set. However, the most popular consensus methods (strict and majority rule) are susceptible to so-called *rogue taxa* [20]. That is, while the tree set may agree very strongly on the structure relating a large subset of the leaves, the remaining few leaves (the rogue taxa) can effectively prevent this underlying structure from appearing in the strict or majority consensus tree. In other words, these methods end up discarding structural elements that are, in fact, prevalent in the set.

Much work has been done on the problem of summarizing a set of trees and on the issue of rogue taxa in particular. The pioneering work of Wilkinson [20,21,22] addresses the problem by returning *sets* of trees, some of which are missing leaves, with the aim of conveying the prevalent structural elements in at least one

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 128–139, 2010.

of the returned trees. While theoretically satisfying, this approach suffers from computational complexity problems and, more importantly, from difficulties in interpretation.

A problem closely related to both consensus and rogue taxa is the *Maximum Agreement Subtree (MAST)*. A MAST on a set of input trees is the subtree of largest leaf-set cardinality common to all input trees. While the general problem of finding the MAST of three or more trees is $\mathcal{NP}$-hard [1], it can be solved efficiently when at least one of the input trees has bounded degree [7]. Another agreement subtree optimization problem *Maximum Information Subtree (MIST)* was proposed by Bryant[3] to overcome a key deficiency of MAST, namely that the maximization of leaf-set cardinality can entirely obscure important internal structure revealed by a smaller, suboptimal for MAST, leaf subset. Bryant's algorithm for solving MIST, whose complexity mirrors that of MAST algorithms, actually affords the practitioner an option to weight the importance placed on leaf-set cardinality versus internal structure in the solution. As such, the optimization function for MIST has a striking resemblance to the MISC optimization problem we propose below. Unfortunately, *all* agreement subtree approaches tend to be too conservative for our purpose; most notably, there exist instances where the strict consensus tree (without dropping any leaves) has more internal edges than any MAST or MIST[13].

Cranston and Rannala recently presented a Markov Chain Monte Carlo (MCMC) method for identifying a version of rogue taxa in the context of Bayesian phylogenetic reconstruction [6]. Their approach identifies subsets of leaves for which the posterior distribution strongly supports the structure of the induced subtree—leaves left out can be viewed as rogue taxa, albeit in the narrow context of a sampling of trees in a Bayesian search, rather than in the general context of a consensus of trees. All of the approaches mentioned thus far fall into the category of "leaf-dropping methods," in the terminology of Redelings [16]. In contrast, Redelings presents, again in the context of Bayesian phylogenetics, a method that returns a "multi-connected tree," which includes all leaves, but does not summarize the information through a single tree and thus again raises issues of interpretation—an issue plaguing all approaches producing non-trees [2,5,10,11].

In this paper we contribute another leaf-dropping method, one based on a rigorous definition of the tradeoff involved between dropping leaves and uncovering additional consensus structure. Most existing measures and methods discard leaves in order to uncover *any* underlying structure; in contrast, our approach sets up a bicriterion problem, in which leaves should be discarded only if the gain in uncovered internal edges outweighs the loss incurred by discarding the leaves. We are not the first researchers to define some notion of relative information content for consensus trees [19], but our definition is the first to both *explicitly* take into account the loss incurred by dropping taxa, and generalize outside the setting of agreement subtrees. We provide an effective greedy heuristic to compute a good (if not necessarily optimal) set of rogue taxa and apply it to both pathological examples from the literature and a collection of large biological datasets that we used in a prior study of bootstrapping. As the presence

of rogue taxa in a set of bootstrap replicates can lead to deceivingly poor support values, we propose a procedure to recompute support values in light of the rogue taxa identified by our algorithm; applying this procedure to our biological datasets caused a large number of edges to change from "unsupported" to "supported" status, indicating that many existing phylogenies should be recomputed and reevaluated to reduce any inaccuracies introduced by rogue taxa.

The rest of the paper is organized as follows. In Section 2 we define concepts and terminology. In Section 3 we define our measure of relative information content, formalize the bicriterion optimization problem for consensus and rogue taxa, and present some theoretical results that underlie our approach. In Section 4 we present an efficient greedy heuristic for our bicriterion problem. In Section 5 we present the results of our experiments.

2 Preliminaries

We use standard set and graph terminology and notation; in particular, $\cup$ refers to union, $\cap$ to intersection, $\setminus$ to set difference, and Δ to symmetric difference—i.e., $S \Delta T = (S \cup T) \setminus (S \cap T)$.

A *phylogenetic tree* represents the evolutionary relationships among a collection of living organisms. Homologous molecular sequences (one for each organism) are placed at the tips of the tree—hereafter called the *leaves*; the internal structure of the tree—its *edges* (sometimes also called branches)—represents the evolutionary relationships. The removal of an edge disconnects the tree and partitions the set of leaves into two subsets; thus each edge corresponds to a *bipartition* of the set of leaves. Every tree includes the same *trivial bipartitions*, which separate one leaf from all others; the other bipartitions are called *nontrivial* and correspond to an *internal* edge of a tree, that is, an edge not incident on a leaf. We can thus view a phylogenetic tree as a leaf-labeled tree $T = (L, B)$, where L is the set of leaves and B is its set of nontrivial bipartitions. To describe a bipartition, we list the two sets of leaves, separated by a | symbol. To ensure an equivalence between nontrivial bipartitions and internal edges, we require that every internal node in a phylogeny have degree at least 3. The number $|B|$ of nontrivial bipartitions in a phylogeny is at most $|L| - 3$; when the two are equal, we say that the (binary) tree is *fully resolved*; otherwise, there must exist an internal node of degree at least 4 and any such node is known as a *polytomy*.

The *consensus* problem is given by a set $\mathcal{T}$ of m trees defined on a common set L of n taxa (leaves). The *bipartition profile* of $\mathcal{T}$ is the pair

$$\mathcal{P} = (B_{\mathcal{T}}, \nu \colon B_{\mathcal{T}} \rightarrow 2^{\mathcal{T}})$$

where $B_{\mathcal{T}}$ is the set of all nontrivial bipartitions found across all m trees in the set and ν is a function mapping bipartitions to the trees in which they appear.

We denote the removal of leaves from trees through the *restriction* operator—which also uses the | symbol. For example, $\mathcal{T}|L'$ refers to restricting each tree in the set $\mathcal{T}$ to the leaf subset $L' \subseteq L$, which corresponds to removing each leaf in $L \setminus L'$ from each tree, as well as removing any nodes of degree 2 created in the

process. Individual trees, tree sets, and tree profiles can appear on the left-hand side of the restriction operator.

We focus on consensus methods based on bipartition *frequency*—see the excellent survey of Bryant [4] for a comprehensive treatment of consensus methods. Given a threshold parameter $\frac{m}{2} < t < m$, the t-consensus tree is composed of all of the bipartitions that occur in more than t trees. The *majority rule* consensus [12] is obtained by setting t to $\frac{m}{2}$, while the *strict* consensus is obtained by setting t to $m-1$. We denote t-consensus methods by $\mathcal{C}_t$. Thus $\mathcal{C}_{m-1}(\mathcal{T})$ corresponds to taking the strict consensus tree of the set $\mathcal{T}$.

3 Relative Information Content, Consensus, Rogue Taxa

3.1 The Measure and the Problem

The general problem we study can be phrased as follows: given a set $\mathcal{T}$ of trees on a common leaf set L and given a frequency-based consensus method $\mathcal{C}_t$, we want to find a leaf subset L' that optimizes the relative information content of the consensus returned by $\mathcal{C}_t$ on the set of subtrees induced by L'. The crucial notion here is that of relative information content. Formally, if $\mathcal{C}_t(\mathcal{T}|L')$ yields $T' = (L', B')$, then the *relative information content* is

$$I(T', L, \mathcal{C}_t) = \frac{|L'| + |B'|}{|L| + (|L| - 3)} \tag{1}$$

This measure is the ratio of the total number of bipartitions (trivial and nontrivial) in the consensus tree derived on the reduced leaf set to the total number of bipartitions in an ideal, fully resolved tree on the original leaf set. By taking trivial bipartitions into account, we automatically penalize a method for removing many leaves, since the number of trivial bipartitions is simply the number of leaves. By adding the number of nontrivial bipartitions, we reward a method for preserving more internal edges, since the denominator is fixed to the number of such edges in an ideal tree. Note that the use of the word 'information' in our definition does not imply information-theoretic foundations.

We can now formulate our main problem, which we call *MISC*, for *Maximum-Information Subtree Consensus.*

Problem. Given a set $\mathcal{T}$ of trees defined on a common leaf set L and a frequency-based consensus method $\mathcal{C}_t$, find a leaf subset L' that maximizes the relative information content $I(\mathcal{C}_t(\mathcal{T}|L'), L, \mathcal{C}_t)$.

Note that the MAST solution typically maximizes the $|B'|$ term at the expense of the $|L'|$ term—it has no direct penalty for dropping leaves; in contrast, consensus methods typically maximize $|L'|$ (in the case of majority and strict consensus, by forcing $L' = L$) at the expense of $|B'|$. MISC, on the other hand, combines the two aspects into a single formulation.

3.2 How Bipartitions Change under Leaf Deletion

We begin by studying the effect that dropping leaves has on a bipartition profile. For any bipartition in the original profile, there are three cases. We illustrate these cases through a simple example, with an original leaf set of a, b, c, d, e, f and with leaves b and e dropped.

1. **merge:** If two bipartitions differ solely in (a subset of) the leaves being dropped, then those bipartitions get merged in the new profile. For example $ac|bdef$ and $abc|def$ merge into $ac|df$ and the ν set for the merged bipartition consists of the union of the two original bipartitions.
2. **disappear:** If dropping the leaves creates a bipartition with an empty side or makes the bipartition trivial, then the bipartition disappears. For example, both $acdf|be$ and $acd|bef$ disappear.
3. **no change:** Otherwise, a bipartition remains unchanged.

An important observation is that, for all $L'' \subseteq L' \subseteq L$, every nontrivial bipartition in $\mathcal{P}|L''$ and in $\mathcal{C}_t(\mathcal{T}|L'')$ arises as a result of a "no change" of a single bipartition or a "merge" of two or more bipartitions in $\mathcal{P}|L'$. Unfortunately this observation does not suggest an efficient algorithm.

3.3 Finding Subsets of Leaves to Drop

Given two bipartitions b_1 and b_2 of L, we can easily identify all leaf subsets L' of minimum cardinality such that dropping L' from L merges b_1 and b_2. If we have $b_1 = A|B$ and $b_2 = C|D$, then the *dropset* L' is the smaller of the two following sets (or either set in case they have the same size):

$$(A \Delta C) \cup (B \Delta D) \text{ or } (A \Delta D) \cup (B \Delta C) \tag{2}$$

This concept is exploited in Algorithm 1.

Observe that, in the terminology of [17], the dropset of b_1 and b_2 is the largest partial X-split such that b_1 and b_2 both extend it.

Algorithm 1. Find minimum cardinality leaf-dropset that renders $b_1 = b_2$

Input: two bipartitions on the same leaf set
Output: the dropset (or dropsets if there are two)

```
1: function BIPARTITION-PAIR-DROPSET(b1 = A|B, b2 = C|D)
2:     S0 ← AΔC ∪ BΔD
3:     S1 ← AΔD ∪ BΔC
4:     if |S0| < |S1| then
5:         return [S0]
6:     else if |S1| < |S0 then
7:         return [S1]
8:     else
9:         return [S0,S1]
10:    end if
11: end function
```

Theorem 1. *Algorithm 1 computes the minimum cardinality dropset for any pair of bipartitions of L.*

Proof. That the dropset causes the two partitions to merge is evident. We establish that the dropset has minimum cardinality by contradiction. Consider that there exists a smaller dropset merging the two bipartitions. Then there is at least one leaf ℓ in the dropset returned by our algorithm that is not in the smaller dropset. This leaf must be on the same side of the partition in both b_1 and b_2, since otherwise our dropset would not merge the two. But our algorithm uses the symmetric difference of these two sides in computing the dropset, so it could not have chosen ℓ, a contradiction. □

Theorem 2. *The cardinalities of the dropsets returned by Algorithm 1 define a metric on the space of bipartitions of L.*

Proof. Three properties characterize a metric: it must be positive definite and symmetric, and it must obey the triangle inequality. The first two properties are trivial in this case. Suppose we have bipartitions b_1, b_2, and b_3; we want to show that the cardinality of the dropset of b_1 and b_3 cannot exceed the sum of the cardinalities of the dropsets of b_1 and b_2 and of b_2 and b_3. Note that removing both of these dropsets from both b_1 and b_3 merges the two bipartitions, thereby establishing an upper bound on the distance between these two bipartitions in our space; but the distance is the size of the dropset of b_1 and b_3, so that the triangle inequality holds. □

4 The Algorithm

We describe the algorithm at a conceptual level, leaving a more formal specification to inset text. First, we build the bipartition profile for the given tree set. Next, we compute the dropset for each pair of bipartitions in the profile such that neither bipartition in the pair appears in the consensus tree, but the pair would appear if merged. For each unique dropset we accumulate the list of bipartition pairs yielding that dropset. These last two parts are formalized in Algorithm 2. We then compute the *impact* of each dropset as the number of bipartition pairs giving rise to that dropset minus the size of the dropset itself. This score corresponds roughly to the difference between the number of edges that will be created and the number of leaves that will be lost should that dropset be used. The dropset of largest impact is then used, the profile updated, the impacts updated, and the process repeated until there does not remain any dropset with a nonnegative impact. This greedy overall framework is formalized in Algorithm 3.

The impact measure ignores disappearing edges and dropsets that are subsets of another—the latter because a superset with deceivingly poor score is likely to get chosen in a subsequent round. The overall algorithm is a greedy heuristic, but does well in practice and on hard instances, as we demonstrate in the next two sections.

There remains the issue, as with all leaf-dropping methods, of what to do with the dropped leaves. The staying power of consensus methods argues for producing a single tree and our method does that. For the rogue taxa, we provide an intriguing strategy that is applicable in some settings in Section 5.3.

5 Experimental Results

We have implemented our approach as a standalone Python-based prototype. Our current implementation is suitable for datasets of up to a thousand trees on a thousand leaves. Scaling up to 10,000 trees on 10,000 leaves is simply a matter of reimplementing our approach as part of RAxML [18] so as to leverage the efficient bipartition manipulation routines therein. In the following, we present results on artificial datasets constructed to cause difficulties to various consensus methods, followed by results on biological datasets that we used in previous work on bootstrapping. We then discuss implications of our results on the interpretation of phylogenetic reconstruction. We conclude by a smaller study on biological datasets using a slight modification of our algorithm to maximize the number of nontrivial bipartitions in the result.

5.1 Difficult Instances

Our algorithm is particularly well suited to the so-called "pathological" instances used in the literature to critique the strict or majority consensus. A classic example is an instance where the trees share a common subtree of $n-k$ leaves, but where the remaining k leaves destroy resolution in the consensus. The example we present here is rather simple and space limits prevent us from giving more examples. We refer interested readers to [13] for expanded treatment of this issue.

Algorithm 2. Find potential dropsets by examining all pairs in a profile

Input: A bipartition profile $\mathcal{P} = (L, B_\mathcal{T}, \nu : B_\mathcal{T} \rightarrow 2^\mathcal{T})$
Input: A frequency-only consensus method $\mathcal{C}_t$ with threshold t
Output: An object mapping dropsets to lists of bipartition pairs

```
1: function POTENTIAL-PROFILE-DROPSETS(P, C_t)
2:     Γ ← {b | b ∈ B_T and |ν(b)| ≤ t}
3:     for all pairs of bipartitions b1,b2 in Γ do
4:         if |ν(b1) ∪ ν(b2)| > t then
5:             L ← BIPARTITION-PAIR-DROPSET(b1, b2)
6:             for d ∈ L do
7:                 δ[d] ← δ[d] ∪ {(b1, b2)}
8:             end for
9:         end if
10:    end for
11:    return δ
12: end function
```

Algorithm 3. Our top level iterative heuristic for finding dropsets

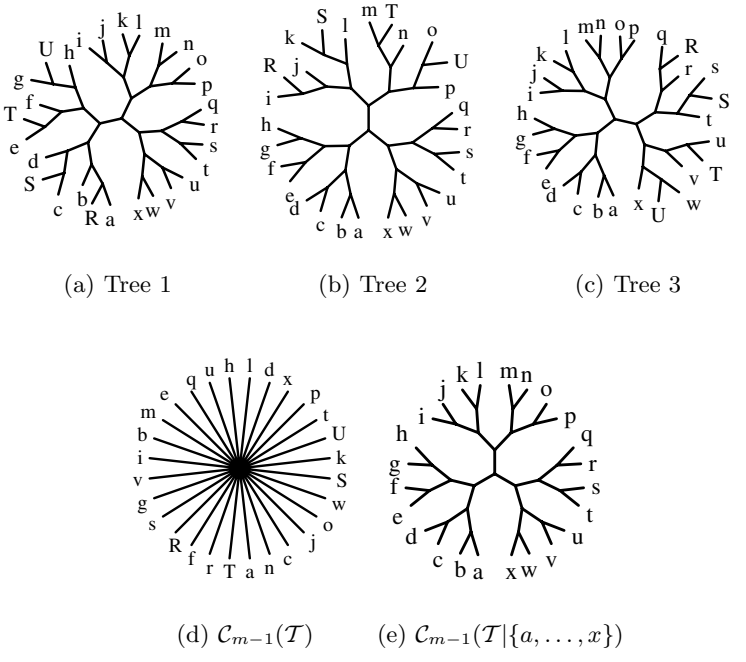
Input: A tree set $\mathcal{T}$
Input: A frequency-only consensus method C with threshold t
Output: A set of leaves to drop, composed of the union of dropsets

function SELECT-AND-REMOVE-DROPSETS($\mathcal{T}$)
  $d^* \leftarrow d_{greedy} \leftarrow \emptyset$
  repeat
    $\mathcal{P} \leftarrow$ BUILD-BIPARTITION-PROFILE($\mathcal{T}|(L - d^*)$)
    $\delta \leftarrow$ POTENTIAL-PROFILE-DROPSETS($\mathcal{P}, \mathcal{C}_t$)
    $maximpact = 0$
    $d_{greedy} = \emptyset$
    for all $d \in \delta$'s domain **do**
      if $|d| - |\delta[d]| \geq maximpact$ **then**
        $d_{greedy} = d$
        $maximpact = |d| - |\delta[d]|$
      end if
    end for
    $d^* = d^* \cup d_{greedy}$
  until $d_{greedy} = \emptyset$
  return d^*
end function

Our example uses the strict consensus. An instance consists of just three trees, defined on the 28-leaf set $\{a, b, \ldots, x, R, S, T, U\}$. The common backbone consists of the 24 taxa $\{a, b, \ldots, x\}$, as illustrated in Figure 1(e)). The rogue taxa form the set $\{R, S, T, U\}$; they vary in position on the backbone as indicated in Figures 1(a), 1(b), and 1(c). The strict consensus tree of the three trees is shown in Fig. 1(d); it is just a star, with no nontrivial bipartition (no internal tree edge) and its relative information content is $I(\mathcal{T}, L, \mathcal{C}_{m-1}) = \frac{28+0}{28+25} = \frac{28}{53} \approx 0.53$. Our algorithm correctly identifies the rogue set, however, so that its strict consensus tree on the remaining set of leaves is the backbone, with an relative information content of $I(\mathcal{T}|\{a, \ldots, x\}, L, \mathcal{C}_{m-1}) = \frac{24+21}{28+25} = \frac{45}{53} \approx 0.85$.

5.2 Results on Biological Data

We applied our method to the datasets we used in an earlier study of bootstrapping methods [14] and available at `http://lcbb.epfl.ch/BS.tar.bz2`. There are 10 datasets of single-gene and multi-gene DNA sequences, with anywhere from 125 to 994 taxa. For each dataset we generated 1,000 bootstrap replicates and applied our algorithm to the resulting trees using both $\mathcal{C}_{\frac{m}{2}}$ and $\mathcal{C}_{m-1}$. Our algorithm found rather diverse dropset sizes across the 10 datasets. The results are depicted in Figure 2, where a quartet of histogram bars are shown for each dataset with a nonempty dropset. The first histogram bar (a negative quantity) denotes how many leaves were dropped, while the second bar (a positive quantity) denotes how many nontrivial bipartitions were uncovered. The third bar is the sum of the first two, simply depicting the net (non-normalized) contribution to relative information content. The final bar is discussed in Section 5.3.



(a) Tree 1 (b) Tree 2 (c) Tree 3

(d) $\mathcal{C}_{m-1}(\mathcal{T})$ (e) $\mathcal{C}_{m-1}(\mathcal{T}|\{a,\ldots,x\})$

Fig. 1. A simple, yet starkly contrasting, example (top) for which the strict consensus returns a star tree, but for which our algorithm correctly identifies the rogue taxa and produces a fully resolved tree (bottom)

5.3 Biological Interpretation

Maximum likelihood phylogenetic analyses are typically conducted in two steps. First the reconstruction proper is performed, yielding a "best tree." Then a number of bootstrap replicate trees are generated, say 500 of them; for each bipartition b in the best tree, its support value is calculated as a normalized count of the number of replicates in which b appears. Researchers tend to consider edges with support lower than 75% as unreliable [8].

If rogue taxa are at work in the replicate set, the support values for certain bipartitions can be deceivingly depressed. To remedy this problem, we propose that Algorithm 3 be applied to the replicate set in order to identify rogue taxa. If a dropset of nonzero size is found, this dropset is then removed from each tree in the replicate set. Finally, the support value is calculated as a normalized count of the replicates in which b' appears such that, if we have $b = A|B$, then, without loss of generality, we have $b' = A' \subseteq A|B' \subseteq B$. In this way, support values in the "best tree" are less susceptible to the deceiving influence of rogue taxa. This approach offers one possible solution to the data display problem of leaf-dropping methods. We still return a single tree on the original leaf set (the "best tree" as reconstructed by an ML method), but support values for individual bipartitions more accurately reflect the underlying replicate data. In our datasets, recomputing support values as suggested above yields very intriguing and promising results.

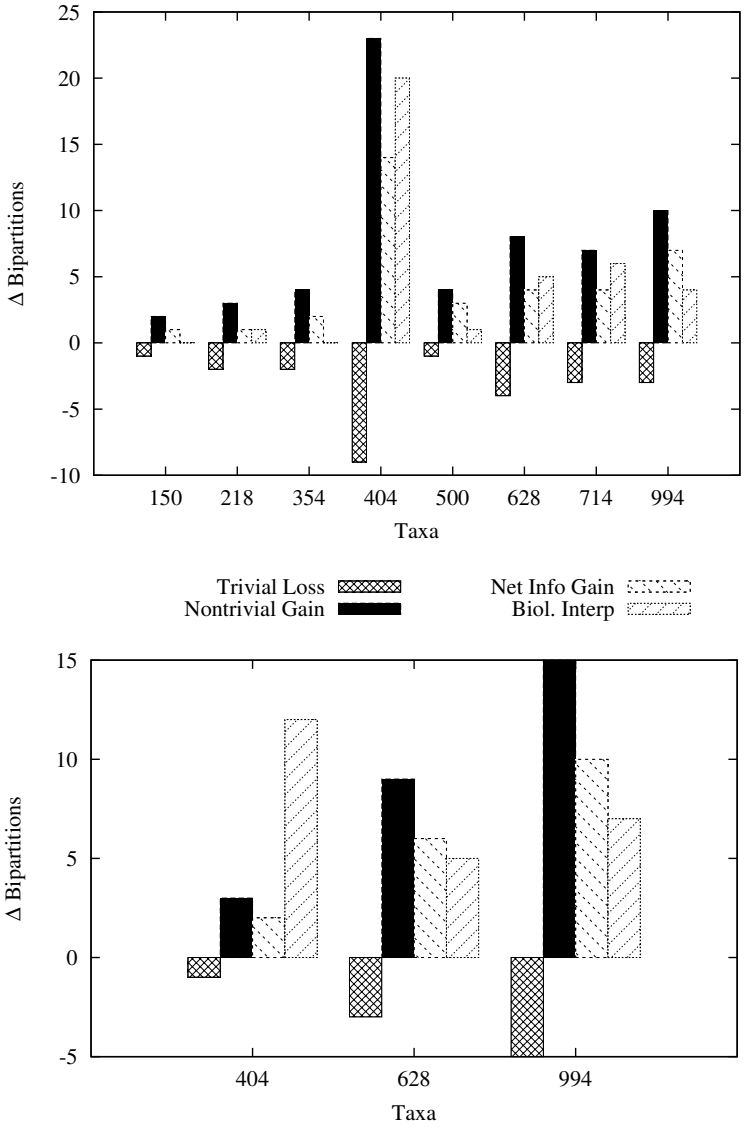


Fig. 2. The performance of Algorithm 3 in terms of how much "hidden" consensus is uncovered in biological data sets. The top plot is for majority consensus, the bottom for strict consensus. The tree sets each consist of 1,000 bootstrap replicates generated by the RAxML 7.2.5 Rapid Bootstrap Algorithm.

All but two of the identified dropsets succeeded in pushing at least one previously hidden edge in the "best tree" over the 75% threshold. The number of edges uncovered by this application of our technique is displayed in the fourth histogram bar in Figures 2(a) and 2(b). In the dataset with 404 taxa, 20 edges were uncovered in this manner, pointing to a need for reevaluation of the phylogeny.

5.4 Increasing Resolution

Our algorithm can easily be modified to maximize nontrivial bipartitions, that is, to remove taxa so as to increase resolution. With such a setting, our algorithm loosely matches the goal of Cranston and Rannala [6], so we analyzed the same dataset with our technique to compare our results to theirs. The data set consists of 85 species of Canformia Carnivora [9]. We obtained the sequence data from TreeBASE (`http://www.treebase.org`, Study Accession # S1532) and reconstructed a tree using RAxML-7.2.5 [18] under the GTRCAT approximation. Additionally, RAxML was used to generate 350 bootstrap replicates (the number chosen by RAxML's bootstopping algorithm). Analyzing these 350 trees with our modified Algorithm 3 and using majority consensus generated fully resolved trees with 50 to 55 taxa, a value consistent with the size of the agreement subtrees observed by Cranston and Rannala [6].

6 Conclusions and Future Work

We have presented a novel framework to define rogue taxa so as to maximize the relative information present in a consensus tree computed after removing these rogue taxa. This framework defines a bicriterion problem, MISC, that is the first to balance explicitly loss of taxa with gain in resolution in a setting other than agreement subtrees. We have also provided an effective greedy heuristic to find a good set of such rogue taxa. This algorithm was tested on both pathological cases from the literature and a variety of biological data. The changes in the consensus tree can be parlayed into more accurate bootstrap scores, which in turn can lead to the reevaluation of phylogenetic trees, as we showed on our biological datasets.

Further work includes a characterization of the computational complexity of the MISC problem, as well as improved algorithms for it, including approximation algorithms with known performance guarantees. Generalizing our approach to support consensus methods other than frequency-based methods is another algorithmic problem worth investigating. Finally, there is certainly room to extend and apply our techniques in different domains, most notably in Bayesian phylogenetics (as suggested in Section 5.4) and for the subtree mergers used in the Disk-Covering Methods (as suggested in [15]). On the bioinformatics side, our preliminary findings indicate that existing phylogenies can be significantly refined by applying our approach to the recomputation of bootstrap support.

Acknowledgements

NDP thanks Alexandros Stamatakis for useful feedback regarding experimental design and for collating the biological datasets used in Section 5.

References

1. Amir, A., Keselman, D.: Maximum agreement subtree in a set of evolutionary trees. SIAM J. Comput. 26, 758–769 (1994)

2. Bandelt, H., Dress, A.: Split decomposition: A new and useful approach to phylogenetic analysis of distance data. Mol. Phyl. Evol. 1(3), 242–252 (1992)
3. Bryant, D.: Hunting for trees, building trees and comparing trees: Theory and method in phylogenetic analysis. PhD thesis, U. Canterbury (1997)
4. Bryant, D.: A classification of consensus methods for phylogenetics. In: Bioconsensus. DIMACS Series in Discrete Math. & Theor. Comput. Sci, vol. 61, pp. 163–184. AMS Press (2002)
5. Bryant, D., Moulton, V.: Neighbor-Net: An agglomerative method for the construction of phylogenetic networks. Mol. Bio. Evol. 21(2), 255–265 (2004)
6. Cranston, K.A., Rannala, B.: Summarizing a posterior distribution of trees using agreement subtrees. Sys. Bio. 56(4), 578–590 (2007)
7. Farach, M., Przytycka, T.M., Thorup, M.: On the agreement of many trees. Inf. Proc. Letters 55(6), 297–301 (1995)
8. Felsenstein, J.: Inferring Phylogenies. Sinauer Assoc. Inc., Boston (2004)
9. Fulton, T.L., Strobeck, C.: Molecular phylogeny of the arctoidea (carnivora): Effect of missing data on supertree and supermatrix analyses of multiple gene data sets. Mol. Phyl. Evol. 41(1), 165–181 (2006)
10. Gauthier, O., Lapointe, F.J.: Seeing the trees for the network: Consensus, information content, and superphylogenies. Sys. Bio. 56(2), 345–355 (2007)
11. Huson, D.: SplitsTree: Analyzing and visualizing evolutionary data. Bioinformatics 14(1), 68–73 (1998)
12. Margush, T., McMorris, F.R.: Consensus n-trees. Bull. Math. Bio. 43, 239–244 (1981)
13. Pattengale, N.D.: Efficient Algorithms for Phylogenetic Post-Analysis. PhD thesis, U. New Mexico (2010)
14. Pattengale, N.D., Alipour, M., Bininda-Emonds, O.R.P., Moret, B.M.E., Stamatakis, A.: How many bootstrap replicates are necessary? In: Batzoglou, S. (ed.) RECOMB 2009. LNCS, vol. 5541, pp. 184–200. Springer, Heidelberg (2009)
15. Pattengale, N.D., Swenson, K.M., Morin, M.M., Moret, B.M.E.: Higher fidelity subtree merging for disk-covering methods. Poster, Algorithmic Biology (2006), `http://www.calit2.net/events/algorithmicbio/files/PattengaleAlgoBio2006.pdf`
16. Redelings, B.: Bayesian phylogenies unplugged: Majority consensus trees with wandering taxa, `http://www4.ncsu.edu/~bdredeli/wandering.pdf`
17. Semple, C., Steel, M.: Tree reconstruction via a closure operation on partial splits. In: Gascuel, O., Sagot, M.-F. (eds.) JOBIM 2000. LNCS, vol. 2066, pp. 126–134. Springer, Heidelberg (2001)
18. Stamatakis, A.: RAxML-VI-HPC: Maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. Bioinformatics 22(21), 2688–2690 (2006)
19. Thorley, J.L., Wilkinson, M., Charleston, M.: The information content of consensus trees. In: Studies in Classification, Data Analysis, and Knowledge Organization, Adv. in Data Science and Classif., pp. 91–98. Springer, Heidelberg (1998)
20. Wilkinson, M.: Common cladistic information and its consensus representation: Reduced Adams and reduced cladistic consensus trees and profiles. Sys. Bio. 43(3), 343–368 (1994)
21. Wilkinson, M.: More on reduced consensus methods. Sys. Bio. 44, 435–439 (1995)
22. Wilkinson, M.: Majority-rule reduced consensus trees and their use in bootstrapping. Mol. Bio. Evol. 13(3), 437–444 (1996)

An Agglomerate Algorithm for Mining Overlapping and Hierarchical Functional Modules in Protein Interaction Networks

Jun Ren[1,2], Jianxin Wang[1], Jianâer Chen[1], Min Li[1], and Gang Chen[1]

[1] School of Information Science and Engineering, Central South University, Changsha, 410083, China
[2] College of Information Science and Technology, Hunan Agricultural University, Changsha, 410128, China
jxwang@mail.csu.edu.cn, renjun19@163.com, jianer@mail.csu.edu.cn

Abstract. Real PPI networks commonly have large size. Functional modules in them are usually overlapping and hierarchical. So it is significant to identify both overlapping and hierarchical modules with low time complexity. However previous methods can not do it. A new agglomerative algorithm, MOMA, is proposed in the paper to resolve this problem. MOMA classifies subgraphs into clusters and vertices. Clusters can overlap each other. MOMA identifies overlapping and hierarchical functional modules by merging overlapping subgraphs. Its time complexity is $O(N^2)$. We apply MOMA, G-N algorithm and Cfinder on the yeast core PPI network. Comparing with G-N algorithm, MOMA can identify overlapping modules. Comparing with Cfinder, MOMA can identify hierarchical modules. Distributions of the lowest P-value show that the module set identified by MOMA has the stronger biological significance than those identified by the other two algorithms.

Keywords: overlapping functional module, hierarchical module structure, agglomerative algorithm.

1 Introduction

Large protein-protein interaction (PPI) databases such as DIP [1], MIPS [2] and SGD [3] have emerged with the development of high-throughput methods. Accumulating evidence suggests that these PPI networks are composed of interacting modules which perform certain biological functions [4-7]. Identifying functional modules in these PPI networks is important to understand the cellular organization and functional mechanisms. Many methods have been proposed to identify functional modules of PPI networks [8-17]. These methods can be roughly classified into two categories.

The first kind of method produces a partition and each vertex belongs to one and only one functional module. Typical algorithms of this method are G-N and G-N modified algorithm [8,9], MoNet [6] and FAG-EC [10,11]. G-N algorithm defines edge betweenness and separates a graph into subgraphs by iteratively removing the

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 140–151, 2010.

edge with the highest betweenness value. Luo proposed MoNet algorithm based on weak modules defined by Radicchi and G-N algorithm [6,12]. Li defined a new local variable, edge clustering coefficient, to replace edge betweenness and proposed FAG-EC algorithm based on G-N algorithm and edge clustering coefficient. As edge clustering coefficient is a local variables, FAG-EC has a low time complexity and can deal with large PPI networks.

The other kind of method identifies functional modules as dense subgraphs. Typical algorithms of this method are CPM/Cfinder [13,14], MCODE [15], DPClus [16]. CPM proposed a clique percolation method (CPM) to mine adjacent k-cliques chains. Cfinder is a famous network analysis tool based on CPM. Both MCODE and DPClus first choose seed vertices by local neighborhood density and then expands seed vertices to density clusters by recursively adding the qualifying neighbor vertices.

Both methods have disadvantages. The functional modules in PPI network are usually overlapping and hierarchical [1-3,13]. The first kind methods can't identify the overlapping modules as they make every vertex belongs to one and only one functional module. The second kind method can't identify the hierarchical functional module as they are not agglomerate algorithms.

In recent years, some authors propose the third kind of method to identify overlapping and hierarchical functional modules. EAGLE is the typical algorithms of this method [17]. EAGLE can identify overlapping and hierarchical functional modules by recursively merging overlapping maximal cliques. However EAGLE has a high time complexity and can't fit in large PPI network as identifying all maximal cliques is a NP-hard problem.

In this paper, a new agglomerative algorithm, MOMA (Mining Overlapping Modules by Agglomerating), is proposed to identify overlapping and hierarchical functional modules in PPI networks. MOMA classifies subgraphs into cluster subgraphs and vertex subgraphs. Cluster subgraphs are clusters and can overlap each other. MOMA identifies functional modules by recursively merging the pair of subgraphs with the maximum clustering coefficient. It can identify hierarchical modules as it is an agglomerate algorithm. It can identify overlapping modules as cluster subgraphs overlap each other. In addition, its time complexity is $O(N^2)$ and can be used in large PPI network, where N is the number of vertices in the network.

As presented above, G-N algorithm is the most famous algorithm of the first kind method as other algorithms based on it. CPM is the typical algorithm of the second kind method and Cfinder is one of the most popular tools based on CPM. So we compare MOMA with G-N algorithm and Cfinder on the yeast core PPI network from the DIP database. MOMA can identify overlapping and hierarchical functional modules but the other two algorithms can not do it. The lowest P-value of a module reflects the biological significance of the module. A module has strong biological significance if its lowest P-value is small. We compare distributions of the lowest P-values of the three algorithms and find that MOMA identifies the most percentage of modules with the lowest P-value$<<10^{-10}$. It means in the three algorithms, the module set identified by MOMA has the most strong biological significance.

2 Method

2.1 Cluster Subgraph

MOMA identifies functional modules by recursively merging subgraphs. The subgraph including only one vertex is named as vertex subgraph, otherwise it is named as cluster subgraph. The cluster subgraph is initialized based on a vertex pair $<v_1, v_2>$. It is composed of v_1, v_2 and their common adjacency vertices if $<v_1, v_2>$ has an edge or the number of their common adjacency vertices is more than or equal to 2. For example, in Fig 1.A, as *<2, 3>* has two common adjacency vertices 1 and 4, the cluster subgraph 1 is composed of vertices {2, 3, 1, 4} based on it though there is no edge of it. As *<6, 7>* has an edge, the cluster subgraph 2 is composed of vertices {6, 7, 8} based on it though there is only one common adjacency vertex of it. However vertices {2, 5, 6} can not compose of a cluster subgraph based on <2, 6> because it has no edge and has only one common adjacency vertex 5. Some vertices do not belong to any cluster subgraph, for example, vertex 5. They are vertex subgraphs. Obviously a cluster subgraph includes at least a triangle or a quadrangle. A vertex subgraph can only make edges with other vertices, otherwise it is included in a cluster subgraph.

Obviously, cluster subgraphs can overlap each other. For example, in Fig 1.B, the cluster subgraph 3 is composed of vertices {9, 10, 12, 13} based on <12, 13>. The cluster subgraph 4 is composed of vertices {10, 14, 11, 13} based on <10, 14>. The edge between <10, 13> is the overlapping edge of them. We discard the cluster subgraph composed of vertices { 10, 13, 14} based on <10,13> as its vertices all from larger cluster subgraph.

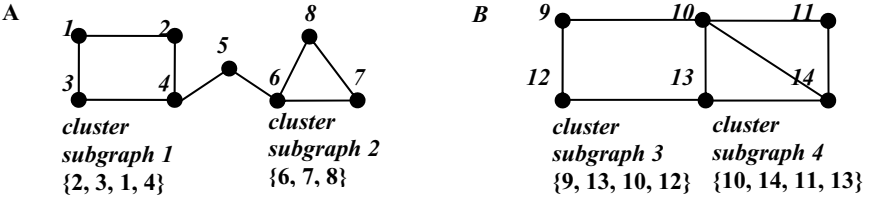


Fig. 1. Cluster subgraph. **A:** initialization cluster subgraph. **B:** overlap of cluster subgraphs.

2.2 Definition of Clustering Coefficient and Module

Every agglomerative algorithm defines a parameter to evaluate the probability of two subgraphs in one module and merges subgraphs according to it. For example, G-N algorithm defines edge betweenness. FAG-EC algorithm defines edge clustering coefficient. As cluster subgraphs overlap each other, MOMA defines a new parameter, clustering coefficient (*CC*) of two subgraphs, to evaluate this probability. To get lower time complexity, *CC* had better be a local variable.

Many evidences show that PPI network is a small world network [18-20]. The *CC* of two vertex subgraphs is equal to 0 as there is no triangle or quadrangle between them. Two cluster subgraphs would have more possibility in one module when there are more overlapping vertices of them and more interaction edges between them. A

vertex subgraph and a cluster subgraph would have more possibility in one module when the vertex connects more vertices in the cluster. Based on these, we define clustering coefficient of two subgraphs as follows:

$$CC_{v1,v2} = 0 \tag{1}$$

$$CC_{c1,c2} = (|E_{over}| + |E_{between}|)/|E_{union}| \tag{2}$$

$$CC_{v1,c1} = |V_{v1\ to\ c1}|/|V_{c1}| \tag{3}$$

where *c1* and *c2* are cluster subgraphs, *v1* and *v2* are vertex subgraphs, $|E_{over}|$ is the number of edges from overlapping vertices to vertices in *c1* or *c2*, $|E_{between}|$ is the number of edges connecting *c1* and *c2* and whose vertices are not overlapping vertices, $|E_{union}|$ is the number of edges of union of *c1* and *c2*, $|V_{v1\ to\ c1}|$ is the number of vertices in *c1* and connecting to *v1*, $|V_{c1}|$ is the number of vertices in *c1*.

Several module definitions have been proposed in literatures [6,10,12]. Radicchi defined the in-degree (k^{in}) of a vertex in an undirected subgraph as the number of edges which connect it to other vertices in the same subgraph and the out-degree (k^{out}) of a vertex as the number of edges which connect it to other vertices in the rest of the graph [12]. In the paper, we define the modularity *M* of a subgraph *C* in a given graph *G* as follows:

$$M_C = \sum_{i \in C} k^{in}(i,C) \Big/ \sum_{i \in C} k^{out}(i,C) \tag{4}$$

where $k^{in}(i,C)$ and $k^{out}(i,C)$ are the in-degree and out-degree of the vertex *i* in the subgraph *C*. Radicchi defined weak module as the subgraph that the sum of in-degree values of its all vertices is larger than the sum of out-degree values of its all vertices [12]. In the paper, we also consider a subgraph is a module when it is a weak module. So a subgraph *C* is a module when $M_C \geq 1$.

2.3 Algorithm MOMA

In the section, a new agglomerative algorithm, MOMA, is proposed. It is described as follows:

1. Initialization cluster subgraphs. In the stage, MOMA first generates *Amatrix* = *G*G*, where *G* is the adjacency matrix of the *PPI* network. *Amatrix*[*i,j*] is the number of common adjacency vertices of the vertex pair <*i,j*>. Select the vertex pair <*i,j* > whose *Amatrix*[*i,j*] is maximum and more than 0. <*i,j*> and their common adjacency vertices compose a cluster subgraph if *Amatrix*[*i,j*]>=2 or <*i,j*> has an edge. The cluster subgraph is added into cluster subgraphs set, *Cvset*, if it has a vertex not belonging to any cluster subgraph in *Cvset*. The time complexity of calculation *Amatrix* is $O(N^2)$, where *N* is the number of vertices in *G*. As each cluster subgraph has at least one vertex not belonging to the other cluster subgraphs, *Cvset* has *N* cluster subgraphs at most and the time complexity of generation *Cvset* is $O(N^2)$. So the whole time complexity of the stage is $O(N^2)$.
2. Every vertex not belonging to *Cvset* is a vertex subgraph. All vertex subgraphs and *Cvset* compose of subgraphs set *Svset*. As each subgraph in *Svset* has at least one vertex not belonging to the other subgraphs, *Svset* has *N* subgraphs at most.

3. Calculation Clustering coefficients. In the stage, MOMA generates *Cmatrix* by calculating the clustering coefficient of each subgraph pair in *Svset* according to the formula (1) , (2) and (3). As *Svset* has *N* subgraphs at most, there are *N*(*N-1*)/2 subgraph pairs at most. So MOMA calculates the clustering coefficient *N*(*N-1*)/2 times at most and the time complexity of the stage is $O(N^2)$.
4. Generation Functional module. In the stage, MOMA generates the functional module set, *Mvset*, by recursively merging the two subgraphs with the maximum *CC* value and are not all modules. After a merging, MOMA recalculates the M_C value of the new subgraph and all *CC* values between it and the other subgraphs connecting with it. Obviously two subgraphs have very little probability in one module when the *CC* value between them is very small. So MOMA defines a parameter *cmin* and ends merging when *CC* values of all subgraph pairs are less than *cmin*. As *Svset* has *N* subgraphs at most, MOMA merges subgraphs *N* times at most and a subgraph connects to *N-1* subgraphs at most. So MOMA calculates *N*(*N-1*) times at most and the time complexity of the stage is $O(N^2)$.
5. Post processing. To every module *m1* in *Mvset*, we can get the module *m2* which has the maximum coverage rate to *m1*. The coverage rate *Cr* is defined as formula (5). In the stage, MOMA merges all <*m1, m2*> whose *Cr* values are more than parameter *max_cr*. Then MOMA deletes all modules with size<3. As *Mvset* has *N* modules at most, the time complexity of generation all <*m1, m2*> is $O(N^2)$.

$$Cr_{m1,m2} = |V_{m1} \bigcap V_{m2}| / \min(|V_{m1}|, |V_{m2}|) \tag{5}$$

where $|V_{m1}|$ is the number of vertices in *m1*, $|V_{m1 \cap Vm2}|$ is the number of overlapping vertices of *m1* and *m2*. As presented above, the time complexity of MOMA is $O(N^2)$.

3 Experiments and Results

We download the yeast core PPI network from the DIP database (version ScereCR 20090106) [1]. Its maximal connected subgraph includes 2092 proteins and 4142 interactions. We implement MOMA, Cfinder and G-N algorithm on it and compare their performance in the section.

3.1 Evaluation of MOMA

P-value is a statistical evaluation criterion which reflects the probability of the co-occurrence of proteins with a given GO annotation in a certain module by chance based on hypergeometric distribution [16,18,21]. P-value is defined as follows:

$$P - value = 1 - \sum_{i=0}^{k-1} \binom{F}{i} \binom{|V| - |F|}{|M| - i} \Big/ \binom{|V|}{|M|} \tag{6}$$

where $|V|$ is the total number of proteins in the network, $|M|$ is the number of proteins in an identified module, $|F|$ is the number of proteins in a real functional module, and *k* is the number of common proteins in the real functional module and the identified

module. Low P-value indicates the identified module corresponds to the real functional module closely because the network has a lower probability to produce the module by chance [22]. So the lowest P-value of a module reflects the biological significance of the module. The smaller the lowest P-value a module has, the stronger biological significance the module has.

In the paper we simplify a real functional module as all proteins participating in a certain biological process. As a biological process is composed of several sub-processes, the structure of real functional modules is hierarchical. The biological process is collected from GO annotations [3,23]. We calculate P-values on GO biological process terms by using the SGD GO Term Finder [3].

Functional modules overlap each other in real biological systems [13]. So we define the overlap rate (*Or*) of a module set, *Mset*, as follows:

$$Or = \sum_{M_i \in Mset} |M_i| / |\bigcup M_i| \tag{7}$$

where $|M_i|$ is the number of proteins in M_i, $|\bigcup M_i|$ is the total number of proteins in *Mset*.

MOMA has two parameters *cmin* and *max_cr*. Tab 1 shows the effect of *cmin* on MOMA when *max_cr*=0.6. Fig 2.A shows distributions of the lowest P-value on GO biological process terms by different *cmin* values when *max_cr*=0.6. As shown in Table 1, with *cmin* decreasing, the number of modules decreases rapidly, the average module size increases rapidly, and the overlap rate decreases slowly because more subgraph pairs can not be merged as their *CC* values less than *cmin*. As shown in Fig 2.A, with *cmin* decreasing, the percentage of modules with the lowest P-value$<10^{-10}$ increases and the percentage of modules with the lowest P-value$>10^{-5}$ decreases. The change is very slowly when *cmin*≤0.25 and becomes rapidly from 0.25. So the biological significance of the module set is decreasing with *cmin* increasing. It decreases slowly when *cmin*≤0.25 and rapidly when *cmin*>0.25.

Table 1. The effect of *cmin* on MOMA when *max_cr*=0.6

Cmin	0.35	0.3	0.25	0.2	0.15	0.1
The number of modules	194	176	155	148	144	139
Average module size	7.35	8.15	9.35	9.98	10.7	11.79
Overlap rate	1.31	1.24	1.20	1.19	1.20	1.19

The lowest P-value and the size of a module are conflicting. It means a smaller module has stronger biological significance than a lager module when the two modules have the same lowest P-value. So the average module size should be considered when using the distribution of the lowest P-value to evaluate the biological significance of a module set. However many algorithms neglect it. As presented above, they are all decreasing with *cmin* increasing. As the average module size decreases rapidly and the biological significance decreases slowly when *cmin*≤0.25, we consider MOMA has the best result when *cmin* = 0.25.

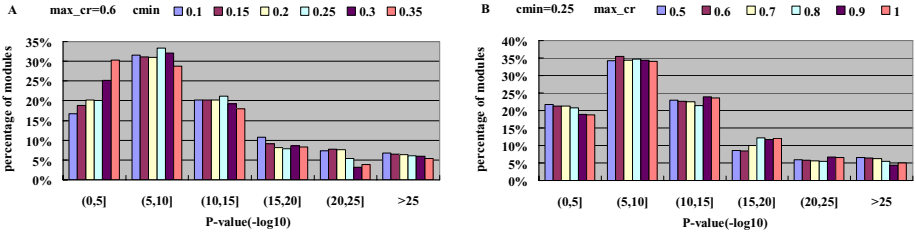


Fig. 2. Distributions of the lowest P-value on GO biological process terms by different *cmin* and *max_c*r values

Table 2 shows the effect of *max_cr* on MOMA when *cmin* =0.25. Fig 2.B shows distributions of the lowest P-value by different *max_cr* values when *cmin*=0.25. As shown in Table 2, with *max_cr* decreasing, the number of modules decreases, the average module size increases, and the overlap rate decreases because more module paris are merged as their *Cr* values more than *max_cr*. However comparing with *cmin*, *max_cr* has much less effort on the result. As shown in Fig 2.B, the distribution also changes little with *max_cr* changing, especially when *max_cr*<=0.6. This is because there are few similar modules identified by MOMA and most of them have *Cr*>0.6.

Table 2. The effect of *max_cr* on MOMA when *cmin*=0.25

max_cr	1	0.9	0.8	0.7	0.6	0.5
The number of modules	182	180	164	160	155	152
Average module size	9.04	9.04	9.26	9.31	9.35	9.48
Overlap rate	1.36	1.34	1.25	1.23	1.20	1.19

3.2 Comparison with G-N Algorithm and Cfinder

We compare MOMA, G-N algorithm and Cfinder in Table 3 and Fig 3. MOMA has two parameters *cmin* and *max_cr*. As analysis in section 3.1, *max_cr* has little effect on the result, especially when *max_cr*<=0.6. So here we set *max_cr* as a constant of 0.6 and adjust *cmin* value. MOMA has the best result when *cmin*=0.25. When *cmin*=0.35, average module sizes of the three algorithms are close. As the module's size effect on its lowest P-value, both MOMA module sets are chose to be compared with the G-N module set and the Cfinder module set. As shown in Table 3, G-N algorithm has the overlap rate equal to 1 and the other algorithms have the overlap rates more than 1. It means that both Cfinder and MOMA can identify overlapping modules but G-N algorithm can't do it.

Table 3. The comparison of modules identified by MOMA, G-N algorithm and Cfinder

	G-N	Cfinder (*k*=3)	Cfinder (*k*=4)	MOMA (*cmin*=0.35)	MOMA (*cmin*=0.25)
The number of modules	269	154	68	194	155
Average module size	7.28	7.08	7.09	7.35	9.35
Overlap rate	1.00	1.15	1.12	1.31	1.20

Fig 3.A shows distributions of the lowest P-values by MOMA, G-N algorithm and Cfinder. As shown in Fig 3.A, in four module sets, MOMA(cmin=0.25), MOMA (cmin=0.35), Cfinder(k=3) and G-N, MOMA(cmin=0.25) has the most percentage of modules with the lowest P-value<10^{-10} and the least percentage of modules with the lowest P-value>10^{-5}. It means MOMA(cmin=0.25) has the strongest biological significance in the four module sets. MOMA(cmin=0.35) has more percentage of modules with the lowest P-value<10^{-10} and less percentage of modules with the lowest P-value>10^{-5} than Cfinder(k=3) and G-N. It means MOMA module set also has stronger biological significance than G-N module set and Cfinder(k=3) module set when they have same average module size. So when k=3, MOMA has the best result in the three algorithms.

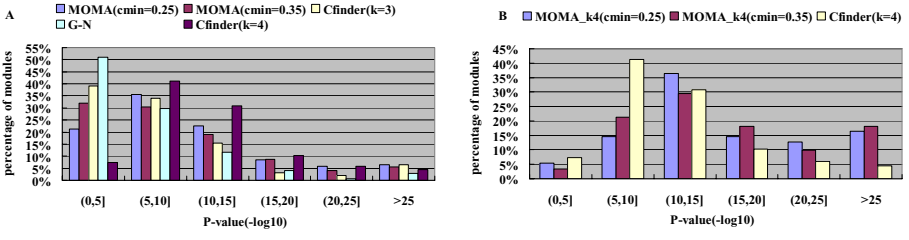


Fig. 3. Distributions of the lowest P-value on GO biological process terms by different algorithms. A: Distributions of the lowest P-value by MOMA, G-N algorithm and Cfinder. B: Distributions of the lowest P-value by Cfinder(k=4) and MOMA_k4.

As shown in Fig 3.A, Cfinder(k=4) identifies the least percentage of modules with the lowest P-value>10^{-5}. However it not means that Cfinder(k=4) has the best result as its number of modules are too small. Table 3 shows that Cfinder(k=4) only identifies 68 modules. It is much less than the other algorithms. All 68 modules can be found in the both MOMA module sets. Their corresponding modules in the MOMA module set compose of the module set MOMA_k4. Table 4 shows that in both MOMA_k4, all 68 modules have at least 75% coverage with their corresponding modules and more than 85% modules of them are completely included in their corresponding modules.

Table 4. Comparison MOMA with Cfinder (k=4)

	Modules with Cr>=0.75		Modules with Cr=1	
	numbers	percentage	numbers	percentage
MOMA_k4 (*cmin*=0.25)	68	100%	61	89.7%
MOMA_k4 (*cmin*=0.35)	68	100%	58	85.3%

Fig 3.B shows the distributions of the lowest P-values by MOMA_k4 and Cfinder(k=4). It shows that the Cfinder(k=4) module set has the least percentage of modules with the lowest P-value<10^{-10} and the most percentage of modules with the lowest P-value>10^{-5}. It means both MOMA_k4 have stronger biological significance than Cfinder(k=4) module set. So both MOMA module sets are better than Cfinder(k=4) module set as every Cfinder(k=4) module is found in them with

Cr>=0.75 and the biological significance of its corresponding modules in both MOMA module sets are commonly stronger than that of itself.

With k value increasing, the number of modules identified by Cfinder is decreasing rapidly. We can get similar result that MOMA has better result than Cfinder(k>4) by the similar analysis presented above. Here we elide it. Conclusion from above, MOMA has the best result in the three algorithms.

Fig 4 and Table 5 give modules identified by the three algorithms to illustrate advantages of MOMA. The seven modules in Fig 4 participate in the same biological process F_1. White vertices are proteins not participating in F_1. MOMA module_2 and MOMA module_120 are shown in left Fig 4. They overlap each other. Proteins with F_1 in MOMA module_2 are colored yellow and those in MOMA module_120 are colored red. Yellow vertex with red bold ring is the overlapping protein of the two modules.

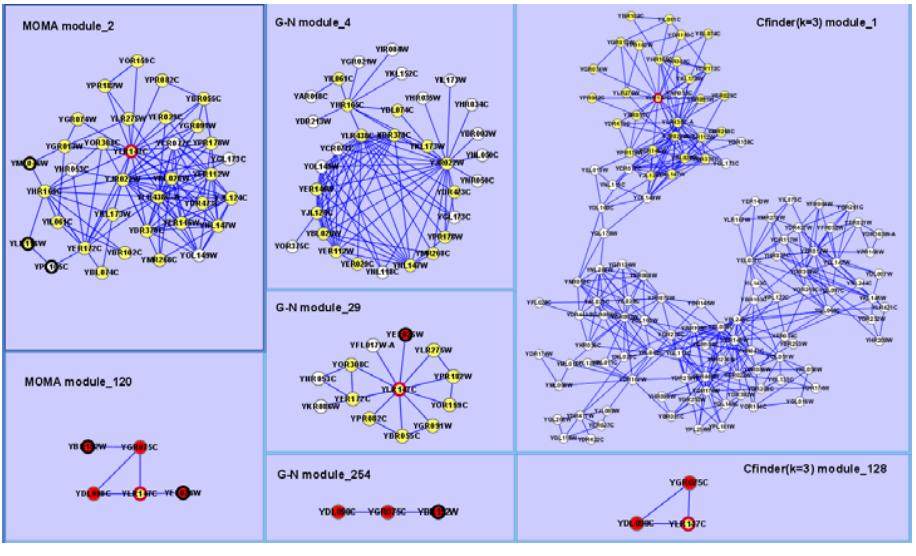


Fig. 4. Compare with modules identified by the three algorithms. Yellow vertices are proteins with F_1 and in MOMA module_2. Red vertices are proteins with F_1 and in MOMA module_120. Yellow vertex with red bold ring is the overlapping protein of MOMA module_2 and module_120. Vertices with black bold ring are peripheral proteins not identified by Cfinder. White vertices are proteins not participating in F_1.

G-N algorithm doesn't allow overlap. MOMA module_2 and MOMA module_120 are divided in three G-N modules which are shown in middle Fig 4. We can see that MOMA module_2 is divided in G-N module_4 and G-N module_29. MOMA module_120 is divided in G-N module_29 and G-N module_254. This division makes G-N modules smaller and more dispersive than their corresponding MOMA modules. So a MOMA module may include several G-N modules with the same biological process, thus its lowest P-value smaller and its biological significance stronger than its corresponding G-N modules. MOMA module_2 has much less white proteins than

G-N module_4. As MOMA modules delete more proteins not belonging to functional modules, their lowest P-values are smaller and their biological significances are stronger than their corresponding G-N modules.

Table 5. A Comparison of modules in Fig 4

Module		$\|M_i \cap F_1\|$	$\|M_i\|$	$\|F_1\|$	P_value
MOMA	Module_2	30	35	103	5.59E-49
	Module_120	5	5	103	2.95E-08
G-N	Module_4	16	33	103	7.13E-19
	Module_29	12	13	103	1.05E-19
	Module_254	3	3	103	1.20E-04
Cfinder (k=3)	Module_1	28	124	103	7.09E-23
	Module_128	3	3	103	7.45E-05

$|M_i|$: protein number of module M_i. $|M_i \cap F_1|$: number of proteins with F_1 and in module M_i.
F_1: RNA splicing, via transesterification reactions with bulged adenosine as nucleophile

Most proteins in MOMA module_2 are in Cfinder(k=3) module_1. Four clusters compose Cfinder(k=3) module_1 and the 28 proteins with F_1 are all in the top cluster. Cfinder(k=3) identifies module by mining adjacent triangle chains. It results the biggest module, Cfinder(k=3) module_1, commonly composed by several adjoining clusters. As MOMA can isolate these clusters, the lowest P-value of MOMA module_2 is much smaller and its biological significance is much stronger than that of Cfinder(k=3) module_1.Cfinder(k=3) module_128 is the core of MOMA module_120. The two red vertices with black bold ring in MOMA module_120 are peripheral proteins. They also participate in biological process F_1 and have important biological significance. However the two peripheral proteins can not be identified by Cfinder (k=3). As MOMA modules neglect less peripheral proteins, their lowest P-values are smaller and their biological significances are stronger than their corresponding G-N modules.

4 Conclusion

Real PPI networks commonly have large size and functional modules in them are usually overlapping and hierarchical. So it is significant to identify both overlapping and hierarchical modules with low time complexity. However previous methods can not do it. In this paper, a new agglomerate algorithm, MOMA, is proposed to solve the problem. MOMA classifies subgraphs into cluster and vertex and defines the clustering coefficient of a subgraph pair. MOMA can identify hierarchical module structure by recursively merging the subgraph pair with the maximum clustering coefficient. It can identify overlapping modules as cluster can overlap each other. It has a polynomial time complexity of $O(N^2)$ and can be used in large PPI networks. We apply MOMA, G-N algorithm and Cfinder on the yeast core PPI network. Distributions of the lowest P-value show that the module set identi-fied by MOMA has the most strong biological significance in the three algorithms

MOMA has disadvantages that it can not select the optimum parameter automatically and can not be used in weighted PPI networks. So we will improve MOMA in the next work.

Acknowledgments. This work is supported in part by the National Natural Science Foundation of China under Grant No.60773111, the Ph.D. Programs Foundation of Ministry of Education of China No. 20090162120073, and the Program for Changjiang Scholars and Innovative Research Team in University No. IRT0661.

References

1. Xenarios, I., Salwínski, L., et al.: DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. Nucleic Acids Res. 30, 303–305 (2002)
2. Mewes, H.W., Frishman, D., Gruber, C., Geier, B., Haase, D., Kaps, A.: MIPS: a database for genomes and protein sequences. Nucleic Acids Res. 28, 37–40 (2000)
3. Issel-Tarver, L., Christie, K.R., Dolinski, K., et al.: Saccharomyces Genome Database. Methods Enzymol. 350, 329–346 (2002)
4. Barabasi, A.L., Oltvai, Z.N.: Network biology: understanding the cell's functional organization. Nat. Res. 5, 101–114 (2004)
5. Chen, J.C., Yuan, B.: Detecting functional modules in the yeast protein-protein interaction network. Bioinformatics 22(18), 2283–2290 (2006)
6. Luo, F., Yang, Y., Chen, C.F., Chang, R., Zhou, J., Scheuermann, R.H.: Modular organization of protein interaction networks. Bioinformatics 23(2), 207–214 (2007)
7. Rives, A.W., Galitski, T.: Modular organization of cellular networks. Proc. Natl. Acad. Sci. USA 100, 1128–1133 (2003)
8. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. USA 99, 7821–7826 (2002)
9. Girvan, M., Newman, M.E.: Finding and evaluating community structure in networks. Phys. Rev. E 69(2), 026113 (2004)
10. Li, M., Wang, J.X., Chen, J.: A Fast Agglomerative algorithm for Mining Functional Modules in Protein Interaction Networks. In: BMEI 2008, pp. 3–7. IEEE press, Los Alamitos (2008)
11. Li, M., Wang, J.X., Chen, J.: Hierarchical organization of functional modules in weighted protein interaction networks using clustering coefficient. In: Măndoiu, I., Narasimhan, G., Zhang, Y. (eds.) ISBRA 2009. LNBIP, vol. 5542, pp. 75–86. Springer, Heidelberg (2009)
12. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proc. Natl.Acad. Sci. USA 101, 2658–2663 (2004)
13. Palla, G., Dernyi, I., Farkas, I.J., Vicsek, T.: Uncoverring the overlapping community structure of complex networks in nature and society. Nature 435(7043), 814–818 (2005)
14. Adamcsek, B., Palla, G., Farkas, I.J., Derényi, I., Vicsek, T.: CFinder: locating cliques and overlapping modules in biological networks. Bioinformatics 22(8), 1021–1023 (2006)
15. Bader, G.D., Hogue, C.: An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinformatics 4, 2 (2003)
16. Altaf-UI-Amin, M., Shinbo, Y., Mihara, K., et al.: Development and implementation of an algorithm for detection of protein complexes in large interaction networks. BMC Bioinformatics 7, 207 (2006)

17. Shen, H., Cheng, X., Cai, K.: Detect overlapping and hierarchical community structure in networks. Physica A 388(8), 1706–1721 (2009)
18. Pržulj, N., Wigle, D.A., Jurisica, I.: Functional topology in a network of protein interactions. Bioinformatics 20(3), 340–348 (2004)
19. Wuchty, S., Almaas, E.: Peeling the yeast protein network. Proteomics 5(2), 444–449 (2005)
20. Yook, S., Oltvai, Z., Barabási, A.: Functional and topological characterization of protein interaction networks. Proteomics 4, 928–942 (2004)
21. King, A.D., Przµlj, N., Jurisica, I.: Protein complex prediction via cost-based clustering. Bioinformatics 20, 3013–3020 (2004)
22. Cho, Y.R., Hwang, W., Ramanathan, M., et al.: Semantic integration to identify overlapping functional modules in protein interaction networks. BMC Bioinformatics 8, 265 (2007)
23. Ashburner, M., Ball, C.A., Blake, J.A., et al.: Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nature Genetics 25, 25–29 (2000)

Fast Protein Structure Alignment

Yosi Shibberu, Allen Holder, and Kyla Lutz

Rose-Hulman Institute of Technology, Department of Mathematics
{Shibberu,Holder,Kyla.Lutz}@rose-hulman.edu

Abstract. We address the problem of aligning the 3D structures of two proteins. Our pairwise comparisons are based on a new optimization model that is succinctly expressed in terms of linear transformations and highlights the problem's intrinsic geometry. The optimization problem is approximately solved with a new polynomial time algorithm. The worst-case analysis of the algorithm shows that the solution is bounded by a constant depending on the data of the problem.

1 Introduction and Background

Proteins play a key role in nearly all biochemical processes of a living organism. The three dimensional structure of a protein molecule largely determines its biological function, and inferences can be made about one protein's function by aligning it to others whose biological function is already established [21]. Hence, protein structure alignment is an important problem in biology.

A protein is a long chain assembled from twenty different types of amino acids called residues. Protein chains fold into unique, tightly packed, globular structures called folds. Typically, a protein's fold is specified by a list of the three dimensional coordinates of each atom in the protein. A distance matrix specifying all the distances between pairs of atoms in the protein completely determines the fold up to reflections in a coordinate invariant way [12]. A distance matrix is often converted into a contact matrix, or map, whose entries equal one for pairs of atoms within a certain cut-off distance from one another and zero otherwise.

The objective in protein alignment is to determine a one-to-one correspondence between a subset of the atoms or residues in two different protein structures. The subset chosen should optimize some biologically relevant similarity measure, although there is currently no consensus on what this measure of similarity should be [21]. In fact, the structure alignment problem itself may not be well-posed in all cases [10].

Existing protein alignment algorithms largely fall into two categories: (i) algorithms that directly use the three dimensional Cartesian coordinates of the atoms and (ii) algorithms that use internal coordinates (e.g. contact matrices) as a basis for comparisons [21]. Unlike sequence alignment, exact polynomial-time structure alignment algorithms do not exist. Kolodny and Linial [18] claim it is possible to obtain an approximate polynomial-time algorithm if one exploits three-dimensional Euclidean geometry. Their claim seems to favor alignment algorithms from category (i). However, three dimensional Euclidean geometry

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 152–165, 2010.

based alignments may introduce undesirable rigidity in the alignment problem [20]. Contact matrix based alignments may be more biologically relevant since they increase flexibility.

The contact map overlap (CMO) protein alignment problem is the problem of determining a one-to-one correspondence between subsets of residues in two proteins that maximizes the overlap of their contact matrices [1,2,6,8,19,22,25]. The CMO problem can be shown to be equivalent to other, well-studied optimization problems, like the maximum subgraph problem [1,2], and is known to be NP-complete [11].

Integer programming formulations of the CMO problem have been solved with branch-and-bound techniques and several associated relaxations [2,6,8,19,25]. The problem was originally formulated in [19] as a binary, quadratic problem. Relaxations of this formulation are studied in [2] and [8], and an exact algorithm is developed in [25]. A fast CMO algorithm that exploits a special structure of the maximum clique problem is described in [22], and a technique that leverages the special properties of self-avoiding walks in two and three-dimensional Euclidean space is developed in [1].

Our approach to protein structure alignment is different. First, we do not use discrete contact maps but instead smooth the contact information and reformulate the problem in n-dimensional Euclidean space, see Figure 1. Second, our geometric reformulation bounds our optimization problem by constructing a solution to the underlying combinatorial problem. Third, integer programming formulations attempt to align proteins using local contact information. We instead take a global perspective by first decomposing the contact maps and identify a smaller collection of characteristic subspaces on which to make alignments. Our method competes favorably with the results in [2] in terms of time and quality, and our algorithm should scale well with problem size.

2 Notation and Problem Statement

Let X be the $n \times 3$ coordinate matrix whose ith row is the coordinates of the ith atom, and let M be the $n \times n$ *distance matrix* whose (i, j) element is the distance between atom i and atom j, i.e.

$$M_{i,j} = \|X_{i,:} - X_{j,:}\|,$$

where $X_{i,:}$ and $X_{j,:}$ are the i-th and j-th columns of X. The matrices X and M are known to be in a one-to-one relationship up to reflection [12]. We let

$$[C(\rho, \kappa)]_{ij} = \max\{\min\{-\rho(M_{i,j} - \kappa), 1\}, 0\},$$

which is a smooth contact matrix, see Figure 2 for a graph of the piecewise linear function. The parameter κ is the distance cutoff parameter and ρ is the magnitude of the slope of the sigmoid. Importantly, if $\rho = 1/\kappa$, then $[C(\rho, \kappa)]_{i,j}$ is arbitrarily small for $i \neq j$ as κ decreases to zero. Hence, we can ensure $C(\rho, \kappa)$ is diagonally dominant and subsequently positive definite. We make this assumption throughout.

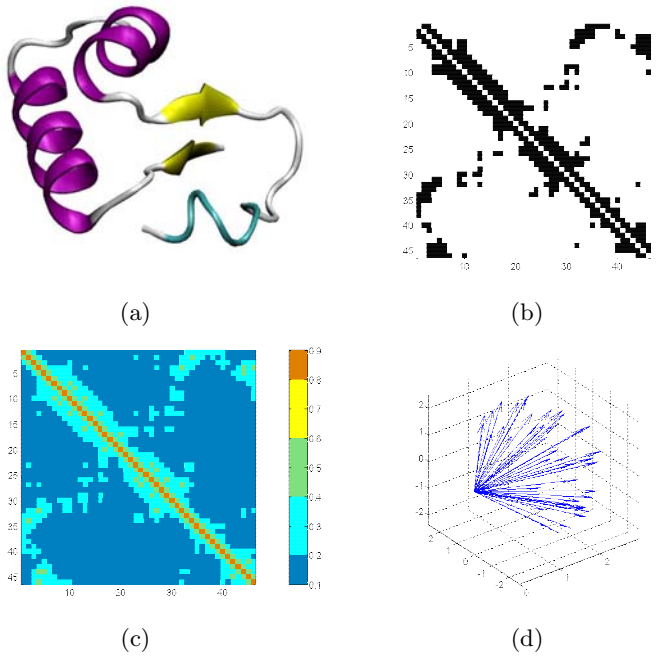


Fig. 1. Representations of the fold of the protein crambin (1crn). (a) 3D representation (b) 8A contact map (c) smoothed positive-definite contact map (d) intrinsic contact vectors projected to R^3.

Let $C'(\rho, \kappa) = C'$ and $C''(\rho, \kappa) = C''$ be contact matrices for two different proteins for which we assume, at least for now, that the number of residues is the same. Although this assumption is atypical, this allows us to succinctly study the fundamentals of our alignment problem, and importantly, it highlights the combinatorial difficulty that we overcome. We adapt our study to the more realistic case of the two proteins having a different number of residues in Section 4.

The assumption that both ρ and κ are selected so that both C' and C'' are positive definite means that there are unitary matrices U and W so that

$$C' = UD'U^T \text{ and } C'' = WD''W^T,$$

where D' and D'' are the diagonal matrices comprised of the positive eigenvalues for C' and C''. Since the eigenspaces and eigenvalues characterize the contact matrices, it makes sense to align them. The essence of our comparison technique rests on the fact that the orthonormality of U and W ensures that we can find a rotation matrix Θ that perfectly aligns U with W, i.e. we can guarantee $\Theta W = U$. However, we have a different rotation for each of the 2^n orientations of the eigenvectors. For example, if we replace the first column of U with its negative, then we have a different rotation. Deciding an optimal rotation means addressing the possibility of searching through all 2^n possible orientations.

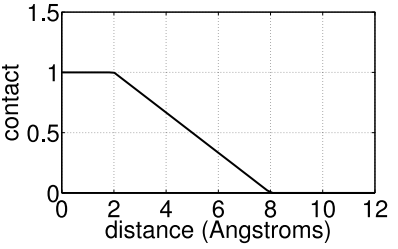


Fig. 2. The graph of $\max\{\min\{-\rho(M_{i,j}-\kappa),1\},0\}$ for $\rho = 1/6$ and $\kappa = 8$. The horizontal axis is $M_{i,j}$ Angstroms.

Three collections of linear operators define our search space, and we let

- $\mathcal{P}$ be the collection of all permutation matrices,
- $\mathcal{R}$ be the collection of all rotation matrices, and
- $\mathcal{I}$ be the collection of all axial reflections, i.e. $\mathcal{I}$ is the set of diagonal matrices $I^{\pm}$ for which each diagonal element is either 1 or -1.

The alignment problem we propose is

$$\min\left\{\|C' - \Theta C''\Omega\|_p^p : \Theta W = UI^{\pm},\ I^{\pm} \in \mathcal{I},\ \Theta \in \mathcal{R},\ \Omega \in \mathcal{P}\right\}. \tag{1}$$

The matrix $I^{\pm}$ orients the eigenvectors of C', for which the unique rotation $\Theta = UI^{\pm}W^T$ aligns the eigenvectors of C'' with those of C'. The permutation matrix Ω pairs the contact vectors to minimize the deviation as measured by the matrix p-norm. We mention that the extreme case in which $\rho \to \infty$ places the problem in graph theoretical terms since both C' and C'' are adjacency matrices for a graph (V, E), with V being the set of respective residues $\{r_1, r_2, \ldots, r_n\}$ and $E = \{(r_i, r_j) : D_{ij} < \kappa\}$. We do not generally consider this case and instead assume throughout that $1 \le p \le 2$ so that the sub-multiplicative property holds.

The problem can be re-written since the constraint $\Theta = UI^{\pm}W^T$ gives

$$C' - \Theta C''\Omega = UD'U^T - UI^{\pm}W^TWD''W^T\Omega = U(D'U^T - I^{\pm}D''W^T\Omega).$$

Using the sub-multiplicative property, we can re-state the problem as

$$\min\left\{\|D'U^T - I^{\pm}D''W^T\Omega\|_p^p : I^{\pm} \in \mathcal{I},\ \Omega \in \mathcal{P}\right\}. \tag{2}$$

Moreover, for the 2-norm we have

$$\begin{aligned}
&\|D'U^T - I^{\pm}D''W^T\Omega\|_2^2 \\
&= \operatorname{tr}\left((D'U^T - I^{\pm}D''W^T\Omega)^T(D'U^T - I^{\pm}D''W^T\Omega)\right) \\
&= \operatorname{tr}\left((U(D')^2U^T - 2UD'I^{\pm}D''W^T\Omega - \Omega^TW(D')^2W^T\Omega\right),
\end{aligned}$$

where $\operatorname{tr}(\cdot)$ is the trace of the matrix. Both $U(D')^2U^T$ and $\Omega^TW(D'')^2W^T\Omega$ are constants under the trace calculation, which means that a 2-norm reformulation is

$$\max\left\{\operatorname{tr}\left(UD'I^{\pm}D''W^T\Omega\right) : I^{\pm} \in \mathcal{I},\ \Omega \in \mathcal{P}\right\}. \tag{3}$$

The 2-norm formulation along with the positive definite assumption provides an intrinsic geometric description of the similarity measure we optimize. Let $R' = \sqrt{D'}U^T$, where the square root is elementwise. We refer to the columns of R' as the *intrinsic contact vectors* of a protein, and each of these corresponds to a residue. Recall that $C'_{i,j}$ is the contact between residue i and residue j. Since $C' = (R')^T R'$, we have that $C'_{i,j} = (R')^T_{:,i} R'_{:,j}$. Moreover, since the diagonal elements of C' are equal to one (every residue is in contact with itself), we have that the intrinsic contact vectors are unit vectors since $C'_{ii} = (R')^T_{:,i} R'_{:,i} = 1$. Therefore, the contact between two residues of a protein is the cosine of the angle between their corresponding intrinsic contact vectors. Allowing $R'' = \sqrt{D''}W^T$, we see that

$$D'U^T = \sqrt{D'}R' \text{ and } D''W^T = \sqrt{D''}R'',$$

and hence, the objective function in (3) is

$$\operatorname{tr}\left(R'\sqrt{D'}I^{\pm}\sqrt{D''}R''\Omega\right).$$

This shows that the 2-norm objective is a scaled sum of the cosines of the angles between the paired intrinsic contact vectors from the two proteins. Since the maximum value of the cosine is 1 if the angle is zero, we have the geometric description that the 2-norm objective is minimizing the angles between the paired intrinsic contact vectors.

3 Algorithmic Motivation

The optimization problem in (2) is a mixed integer optimization problem (MIP), for which a number of exact algorithms are known. However, the binary search tree underlying the MIP formulation has 2^n leaves, each of which corresponds to a unique $I^{\pm}$ in $\mathcal{I}$. For any one of these an optimal permutation matrix Ω can be calculated by solving a traditional assignment problem on the bipartite graph (N', N'', E), where N' is the collection of column vectors in $D'U^T$, N'' is the collection of column vectors in $I^{\pm}D''W^T$, $E = N' \times N''$, and each edge is weighted with the p-norm difference of the defining vectors. While the assignment problem is polynomial, the fact that we might have to solve 2^n of these problems is cause for concern since n is typically around a 100. To test the ability of stock solvers we formed the MIP in AMPL and tried to solve a 10 residue problem with MINLP (posted at NEOS, `http://www-neos.mcs.anl.gov/`). The solution was known to be $I^{\pm} = \Omega = I$. However, MINLP reported a different optimal solution with an objective value about 10 times that of the known optimum. As a counterpart, CPLEX correctly identified the solution by solving the standard MIP relaxation. Unfortunately, similar success for larger, and more difficult, problems was not observed with CPLEX. This demonstrates the need for quick, high-quality heuristics to align large proteins, and we present a new, polynomial-time search strategy based on a geometric bound.

A small example highlights that the assignment problem is bounded for each $I^{\pm}$. The following are from 3 atoms of a beta sheet in two different proteins,

$$D'U^T = \begin{bmatrix} 0.0066 & -0.0128 & 0.0066 \\ 0.0953 & -0.0002 & -0.0950 \\ 1.6278 & 1.6793 & 1.6281 \end{bmatrix}$$

and

$$D''W^T = \begin{bmatrix} 0.0036 & -0.0070 & 0.0036 \\ 0.0104 & -0.0002 & -0.0103 \\ 1.6223 & 1.6819 & 1.6225 \end{bmatrix}.$$

We construct $I^{\pm}$ by minimizing the maximum magnitude of each row of $D'U^T - I^{\pm}D''W^T\Omega$. For example, if the first diagonal element of $I^{\pm}$ is 1, then the maximum magnitude element of the first row of $D'U^T - I^{\pm}D''W^T\Omega$ is

$$\begin{aligned} 0.0194 = &\max\{0.0066, -0.0128, 0.0066, 0.0036, -0.0070, 0.0036\} \\ &- \min\{0.0066, -0.0128, 0.0066, 0.0036, -0.0070, 0.0036\}. \end{aligned}$$

If the first diagonal element of $I^{\pm}$ is instead -1, the maximum magnitude element of the first row of $D'U^T - I^{\pm}D''W^T\Omega$ is

$$\begin{aligned} 0.0198 = &\max\{0.0066, -0.0128, 0.0066, -0.0036, 0.0070, -0.0036\} \\ &- \min\{0.0066, -0.0128, 0.0066, -0.0036, 0.0070, -0.0036\}. \end{aligned}$$

Since the first is lower, we let the first diagonal element of $I^{\pm}$ be 1. For the second diagonal element we find that the maximum possible magnitude difference in the second row is 0.2071 if we choose either 1 or -1, which leaves this element undecided. For the third diagonal element we have a maximum possible magnitude difference of 0.0596 for 1 and 3.3612 for -1, and we select the 1 over the -1. This leaves two choices for the diagonal elements of $I^{\pm}$, either $(1, 1, 1)$ or $(1, -1, 1)$.

This construction of $I^{\pm}$ guarantees the magnitude of the difference between each matrix coefficient of $D'U^T - I^{\pm}D''W^T\Omega$ is at most the corresponding row value independent of Ω. So, for either of our two choices of $I^{\pm}$ we have for any permutation matrix Ω that

$$\left|D'U^T - I^{\pm}D''W^T\Omega\right| \leq \begin{bmatrix} 0.0194 & 0.0194 & 0.0194 \\ 0.2071 & 0.2071 & 0.2071 \\ 0.0056 & 0.0056 & 0.0056 \end{bmatrix},$$

where the absolute value of the matrix is componentwise. This bounds the optimal value of (2) by $3\|(0.0194, 0.2071, 0.0056)^T\|_p^p$, which for $p = 2$ is 0.1299. This problem's unique optimal solution has both $I^{\pm}$ and Ω being the identity, with the optimal value being 0.0001. So the technique identifies the optimal $I^{\pm}$. Importantly, the technique also identifies the two $I^{\pm}$ matrices with the lowest objective values, which are listed in Table 1 for all $I^{\pm}$ and Ω possibilities. The

Table 1. The first column lists the diagonal elements of $I^{\pm}$, so the diagonal of $I^{\pm}$ for the second row is $(1, 1, -1)$. The first row shows the permutation used to order the columns of the identity to form Ω. So, Ω for the second column has the 2nd and third columns of the identity swapped. For ease of presentation the values are rounded to four decimal places, which leaves two values at 0.0001. However, the top, left most value is lower with increased accuracy.

$I^{\pm} \setminus \Omega$	(1,2,3)	(1,3,2)	(2,1,3)	(2,3,1)	(3,2,1)	(3,1,2)
(1, 1, 1)	0.0001	0.0263	0.0266	0.0592	0.0790	0.0591
(1, 1, -1)	32.4270	32.4210	32.4210	32.4210	32.4270	32.4210
(1, -1, 1)	0.0790	0.0594	0.0591	0.0264	0.0001	0.0264
(-1, 1, 1)	0.0006	0.0258	0.0265	0.0594	0.0790	0.0594
(1, -1, -1)	32.4270	32.4210	32.4210	32.4210	32.4270	32.4210
(-1, 1, -1)	32.4270	32.4210	32.4210	32.4210	32.4270	32.4210
(-1, -1, 1)	0.0790	0.0595	0.0593	0.0260	0.0006	0.0260
(-1, -1, -1)	32.4270	32.4210	32.4210	32.4210	32.4270	32.4210

calculation identifying the third diagonal element of $I^{\pm}$ hints that there is possibly a relatively large assignment if -1 is selected. Table 1 shows that the best assignment if the third diagonal is -1 is $O(10^4)$ above the assignments in which the third diagonal is 1.

From a geometric perspective the construction of $I^{\pm}$ orients, or signs, the axial components of the column vectors of $D''W^T$ so that they collapse into the smallest "box" that also contains the column vectors of $D'U^T$. This box bounds the worst possible assignment. Formally, for $\eta_i \in \{1, -1\}$ we let

$$\delta_i^{\min}(\eta_i) = \min_j \left(\{\lambda_i' U_{j,i}\} \cup \{\eta_i \lambda_i'' W_{j,i}\}\right)$$

and

$$\delta_i^{\max}(\eta_i) = \max_j \left(\{\lambda_i' U_{j,i}\} \cup \{\eta_i \lambda_i'' W_{j,i}\}\right).$$

Then setting $\Delta_i(\eta_i) = \left(\delta_i^{\max}(\eta_i) - \delta_i^{\min}(\eta_i)\right)$, we have

$$\max_{\Omega \in \mathcal{P}} \left\{ \|D'U^T - I^{\pm} D'' W^T \Omega\|_p^p : I_{i,i}^{\pm} = \bar{\eta_i} \ \forall i \right\} \leq n \sum_i \min\{\Delta_i(1), \Delta_i(-1)\},$$

where $\bar{\eta_i}$ satisfies $\Delta_i(\bar{\eta_i}) = \min\{\Delta_i(1), \Delta_i(-1)\}$. Since the particular $I^{\pm}$ used here is only one of the 2^n elements of $\mathcal{I}$, we have the following

Theorem 1. *The optimal value of the alignment problem in (1) is no worse than* $n \sum_i \min\{\Delta_i(1), \Delta_i(-1)\}$.

Since calculating all Δ_is is $O(n^2)$, Theorem 1 gives a polynomial upper bound on the problem. Our experimental results show that this bound is not generally indicative of the optimal value of (1), especially if the proteins align well. This is not surprising since the bound is a worst case estimate of the geometry of the problem, and in the case that the proteins align well, the geometric bound is

expected to be a poor estimate of the alignment problem. Indeed, if the proteins align perfectly, then the optimal value is zero while the geometric bound is $\sum_{i=1}^{n} \lambda'_i$, provided that U is the identity. However, there is significant value in calculating the bound since it identifies meaningful orientations. For example, suppose that $\Delta_i(1) << \Delta_i(-1)$. This suggests a preference to sign the i-th eigenvector with a 1 since if we instead select -1, the column vectors of $I^{\pm} D'' W^T$ deviate from the column vectors of $D' U^T$. Since our goal is to minimize deviation, we select 1.

4 Adaptations for Real Numerical Studies

The previous section presents a method of calculating $I^{\pm}$ so that the assignment problem is bounded geometrically, and in this section we develop a polynomial time solution procedure based on this calculation. We first adapt our model to the more realistic case in which

- the number of residues differs between the two proteins, and
- residues from like secondary structures are aligned.

We assume for convenience that the protein with the fewer number of residues corresponds to C'. In this case we pad C' with rows and columns of zeros to the right and to the bottom so that its dimensions agree with C''. Unlike the simplified case studied earlier, part of the alignment problem is to select the eigenvectors of the larger protein that best align with the smaller protein. Let there be n_1 residues in the smaller protein and n_2 in the larger. The required selection is accomplished by a linear operator of the form

$$\Gamma = \begin{bmatrix} \Gamma' \\ \cdots \\ 0 \end{bmatrix},$$

in which Γ' is a $n_1 \times n_2$ binary matrix whose row sums are 1. This alters (2) to become

$$\min \left\{ \| D' U^T - I^{\pm} \Gamma D'' W^T \Omega \|_p^p : I^{\pm} \in \mathcal{I},\ \Omega \in \mathcal{P},\ \Gamma \in \mathcal{G} \right\}, \tag{4}$$

where $\mathcal{G}$ is the collection of all possible Γ matrices. To account for secondary structure alignment we enforce additional restrictions on Ω. Structural motifs, such as β-sheets and α-helices, are identified by the DSSP algorithm from [15], and part of the alignment problem is to encourage the alignment of residues between like structures in the two proteins. Ensuring such alignments can be accomplished by altering Ω. In particular, we can assume that $\Omega_{i,j} = 0$ if the secondary structure of residue i in the first protein disagrees with the secondary structure of residue j in the second protein. Since the number of residues in like secondary structures typically varies between the two proteins, we can no longer

ensure that that each row and column of Ω contains a single 1, and instead, we can only ensure

$$\sum_i \Omega_{i,j} \leq 1 \text{ for all } j, \quad \sum_j \Omega_{i,j} \leq 1 \text{ for all } i, \quad \text{and} \quad \sum_{i,J} \Omega_{i,j} \leq S. \tag{5}$$

The maximum value of S that the summation in the last condition can achieve is the total number of residues that are in a common secondary structure. For example, if the first protein has 8 residues in an α-helix and 3 residues in a β-sheet, whereas the second protein has 5 residues in an α-helix and 4 in a β-sheet, then the maximum value of S that can be achieved is $\min\{8,5\} + \min\{3,4\} = 8$. Since $\sum_{i,j} \Omega_{i,j}$ is the number of paired residues, we generally want this to be large. If we let $\mathcal{P}'$ be the altered set of Ω matrices, the updated alignment problem is

$$\min\left\{\|D'U^T - I^{\pm}\Gamma D''W^T\Omega\|_p^p : I^{\pm} \in \mathcal{I},\ \Omega \in \mathcal{P}',\ \Gamma \in \mathcal{G}\right\}, \tag{6}$$

which can be re-written in terms of the contact matrices as

$$\min\left\{\|C' - \Theta W\Gamma W^T C''\Omega\|_p^p : \Theta W = UI^{\pm}, I^{\pm} \in \mathcal{I},\ \Omega \in \mathcal{P}',\ \Gamma \in \mathcal{G}\right\}.$$

The only interpretive differences between this and (11) are that $W\Gamma W^T$ projects C'' onto a smaller dimension so that it can be aligned with C' and that $\mathcal{P}'$ is altered from $\mathcal{P}$. As discussed momentarily, both Γ and Ω can be calculated efficiently, which means the combinatorial difficulty remains with calculating $I^{\pm}$. Our algorithmic structure circumvents the combinatorial issue of the problem by calculating the Δ_i's as follows,

1. Calculate Γ with an assignment problem.
2. Use $\Gamma D''$ instead of D'' to calculate Δ_i and let

$$I^{\pm}_{i,i} = \begin{cases} 1, \Delta_i(1) < \Delta_i(-1) \\ -1, \Delta_i(1) > \Delta_i(-1) \\ 0, \Delta_i(1) = \Delta_i(-1). \end{cases}$$

3. Calculate Ω with either an assignment problem or dynamic programming.

The fact that $I^{\pm}_{i,i}$ can be zero means that $I^{\pm}$ is acting as an additional projection, i.e. the product $I^{\pm}\Gamma$ is selecting a collection of eigenvectors as well as signing those that are selected. From the previous example we see that the additional projection identifies the coordinates for which the calculation of Δ_i indicates an orientation of the eigenvector. So the combined effect of $I^{\pm}\Gamma$ is to judiciously orient and select the eigenspaces on which to pair the residues.

A traditional assignment problem can be used to calculate one or both of Γ and/or Ω. If we let $\xi_{i,j}$ be the "cost" of assigning entity i to entity j, the classical assignment problem for a square ξ matrix is

$$\min\left\{\sum_{i,j} \xi_{i,j}\omega_{i,j} : \sum_j \omega_{i,j} = 1\ \forall i,\ \sum_i \omega_{i,j} = 1\ \forall j,\ \omega_{i,j} \in \{0,1\}\right\}. \tag{7}$$

To compute Γ we let $\xi_{i,j} = |\lambda'_i - \lambda''_j|$, which encourages eigenvectors with similar eigenvalues to be paired. Since the proteins are of different sizes, we replace $\sum_j \omega_{i,j} = 1$ with $\sum_j \omega_{i,j} \leq 1$. As with the square case, solving the problem is well known to be polynomial. We used the Hungarian algorithm in [5] to calculate Γ. To calculate Ω we let

$$\xi_{i,j} = \|[D'U^T]_{:,i} - [I^{\pm}\Gamma D''W^T]_{:,j}\|_p^p.$$

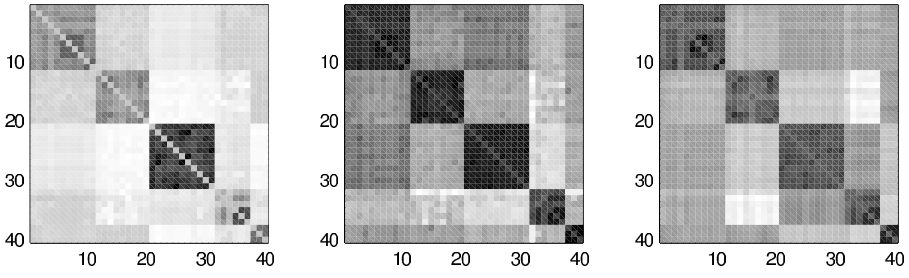
We further replace $\sum_i \omega_{i,j} = 1$ with $\sum_i \omega_{i,j} \leq 1$ and add $\sum_{i,j} \omega_{i,j} = S$, where S the maximum value in (5). This problem was modeled in AMPL and solved with CPLEX due to the changed constraints. Assignment problems were similarly used in [24].

The residue pairings from our initial numerical effort were disappointing in their biological measures. The problem was in the use of the assignment problem to calculate Ω, which was inadequate in its flexibility to handle gaps in the residue pairing. Gaps are controlled by S in the assignment problem. We used the equality $\sum_{i,j} \omega_{i,j} = S$, with S being the largest possible value, to guarantee a match between as many residues as possible. However, this assumption is not biologically sound. As an alternative, we compared the assignment method with a dynamic programming (DP) approach that pairs the residues. The DP algorithm is a standard global sequence alignment procedure [13] that allows, but penalizes, gaps in the alignment. This permits S to deviate from its maximum value. Secondary structure mismatches are also allowed but penalized. We refer readers to [13] for a description of the procedure.

5 Numerical Results

We tested our algorithm's ability to identity the known families identified by SCOP [3] among 33 protein structures taken from the Skolnick data set [2,6], see Table 2. The protein structures in the Skolnick data set were obtained from the Protein Data Bank [4] and parsed with BioPython [7]. The contact matrices were constructed with the piecewise linear sigmoid function mentioned in Section 2 with $\rho = \kappa = 7$. Other sigmoid functions were tested, but the piecewise linear function worked well with these parameters. Both the assignment method and the DP method were tested to calculate the permutation matrix Ω. The RMSD scores of our residue pairings were consistently worse for the assignment method, with an average improvement of 6.2% with DP in both the 1 and 2-norm objectives. For this reason the results below are based on the DP method for calculating Ω. Each gap in the residue alignment was penalized with a value of 2, and pairing residues from different secondary structures was penalized with a value of 3.5. These parameters can be altered to remove/limit either gaps or mismatches. In our numerical work these values gave a mixture of gaps and mismatches.

Our algorithm was run on a dual core 2.16 GHz T2600 Intel processor with 1GB of memory in Matlab under Linux. The algorithm took 555.76 seconds to align 780 pairs of proteins with the 2-norm and 734.59 seconds with the 1-norm,



(a) STRUCTAL (b) RMSD (c) DP Values

Fig. 3. Various scores for our alignments of the Skolnick data set with the 2-norm version of our objective function. The 40 proteins compared are ordered as they are listed in Table 2.

Table 2. The Skolnick Data Set

SCOP Fold	SCOP Family	Proteins
Flavodxin-like	CheY-related	1b00, 1dbw, 1nat, 1ntr, 3chy 1qmp(A,B,C,D), 4tmy(A,B)
Cupredoxin-like	Plastocyanin azurin-like	1baw, 1byo(A,B), 1kdi, 1nin 1pla, 2b3i, 2pcy, 2plt
TIM beta/alpha-barrel	Triosephosphate isomerase (TIM)	1amk, 1aw2, 1b9b, 1btm, 1hti 1tmh, 1tre, 1tri, 1ydv, 3ypi, 8tim
Ferritin-like	Ferritin	1b71, 1bcf, 1dps, 1fha, 1ier, 1rcd
Microbial ribonuclease	Fungal ribonucleases	1rn1(A,B,C)

approximately 0.71 seconds and 0.94 seconds per alignment, respectively. Andonov et al. [2] report a time of approximately 1.04 seconds per alignment for their algorithm on a 2.4 GHz AMD Opteron processor with 4 GB of memory programmed in C++. Computation of the eigensystem of each protein is not included as this is a one time operation. For small proteins the cost of computing the eigensystem of the protein's contact matrix is negligible, but the cost grows quickly for larger proteins. The eigensystems for large proteins should be computed and stored prior to comparison.

The graphs in Figure 3 depict the clustering ability of three different scores of our alignments with the 2-norm objective function. The first two scoring functions are widely used to assess protein alignments. STRUCTAL [23] has been reported to be a good scoring function for protein alignment [17] and is given by

$$\mathsf{STRUCTAL} = \sum_i \frac{20}{1 + \left(\frac{d_i}{2.24}\right)^2} - 10n_g.$$

The quantity d_i is the distance (after the structures have been superimposed) between the ith paired residues/atoms. The quantity n_g equals the total number of gaps in the alignment. Figure 3(a) is a graph of our STRUCTAL scores. The

second scoring function is the RMSD of the aligned residues [14,16], which is shown in Figure 3(b). Figure 3(c) is the score from the DP construction of Ω. Our algorithm correctly identifies the known 5 families in each case. RMSD most clearly distinguishes the families, with our DP values doing nearly as well. The STRUCTAL measure correctly identifies the families, although the delineations are not as sharp (especially for the 4th group).

6 Conclusion

The eigensystem-based protein structure alignment algorithm described in this article is a new and fast way to align protein structures. The geometry of aligning the intrinsic contact vectors of two proteins provides additional insight into the protein alignment problem. This geometric interpretation of the problem is not available from the contact map overlap problem formulation which has a more graph-theoretic flavor. By solving an assignment problem, we can quickly pair the eigenvalues of the contact matrices of two proteins. Once an orientation for the second protein's eigenvectors is specified, the corresponding eigenvectors are easily paired, providing a quick, permutation independent way to compare two protein structures. The key challenge solved in this paper is a method for quickly identifying a good orientation for the eigenvectors of the second protein. The last step in the alignment is to solve a standard global sequence alignment problem. Because this alignment is done only once, the algorithm is fast, at least comparable in speed to the latest algorithms for the contact map overlap problem, but with the potential to scale well for larger problems. Several variants of the approach are possible, and in future work we hope to compare these variants against many of the existing alignment algorithms.

Acknowledgments

The idea of using the eigensystem of protein contact maps to align protein structures was inspired by the work of Galaktionov and Marshall [9] on protein structure prediction. Some preliminary ideas for the algorithm described in this paper were developed during the first author's 2005-06 sabbatical visit, hosted by Garland Marshall, at the Center for Molecular Design, Washington University, St. Louis, Missouri, USA.

References

1. Agarwal, P.K., Mustafa, N.H., Wang, Y.: Fast molecular shape matching using contact maps. J. Comput. Biol. 14(2), 131–143 (2007)
2. Andonov, R., Yanev, N., Malod-Dognin, N.: An efficient lagrangian relaxation for the contact map overlap problem. In: Crandall, K.A., Lagergren, J. (eds.) WABI 2008. LNCS (LNBI), vol. 5251, pp. 162–173. Springer, Heidelberg (2008)
3. Andreeva, A., Howorth, D., Chandonia, J.-M., Brenner, S.E., Hubbard, T.P., Chothia, C., Murzin, A.G.: Data growth and its impact on the scop database: new developments. Nucleic Acids Res. 36(Database issue), D419–D425 (2008)

4. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Res. 28(1), 235–242 (2000)
5. Cao, Y.: Hungarian algorithm for linear assignment problems, v2.1 (2008), http://www.mathworks.com/matlabcentral/fileexchange/20652
6. Caprara, A., Carr, R., Istrail, S., Lancia, G., Walenz, B.: 1001 optimal pdb structure alignments: integer programming methods for finding the maximum contact map overlap. J. Comput. Biol. 11(1), 27–52 (2004)
7. Cock, P.J.A., Antao, T., Chang, J.T., Chapman, B.A., Cox, C.J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., de Hoon, M.J.L.: Biopython: freely available python tools for computational molecular biology and bioinformatics. Bioinformatics 25(11), 1422–1423 (2009)
8. Forrester, R.J., Greenberg, H.J.: Quadratic binary programming models in computational biology. Algorithmic Operations Research 3, 110–129 (2008)
9. Galaktionov, S.G., Marshall, G.R.: Prediction of protein structure in terms of intraglobular contacts: 1d to 2d to 3d. In: Fourth International Conference on Computational Biology, Intelligent Systems for Molecular Biology 1996, St. Louis, Missouri, U.S.A., June 12–15 (1996)
10. Godzik, A.: The structural alignment between two proteins: is there a unique answer? Protein Sci. 5(7), 1325–1338 (1996)
11. Goldman, D., Istrail, S., Papadimitriou, C.H.: Algorithmic aspects of protein structure similarity. In: 40th Annual Symposium on Foundations of Computer Science, pp. 512–521 (1999)
12. Havel, T.F., Kuntz, I.D., Crippen, G.M.: The combinatorial distance geometry method for the calculation of molecular conformation. i. a new approach to an old problem. J. Theor. Biol. 104(3), 359–381 (1983)
13. Jones, N.C., Pevzner, P.A.: An Introduction to Bioinformatics Algorithms. MIT Press, Cambridge (2004)
14. Kabsch, W.: A discussion of the solution for the best rotation to relate two sets of vectors. Acta Crystallog. A 34, 827–828 (1978)
15. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22, 2577–2637 (1983)
16. Kearsley, S.K.: On the orthogonal transformation used for structural comparisons. Acta Crystallogr. A 45, 208–210 (1989)
17. Kolodny, R., Koehl, P., Levitt, M.: Comprehensive evaluation of protein structure alignment methods: scoring by geometric measures. J. Mol. Biol. 346(4), 1173–1188 (2005)
18. Kolodny, R., Linial, N.: Approximate protein structural alignment in polynomial time. Proc. Natl. Acad. Sci. USA 101(33), 12201–12206 (2004)
19. Lancia, G., Carr, R., Walenz, B., Istrail, S.: 101 optimal pdb structure alignments: A branch-and-cut algorithm for the maximum contact map overlap problem. In: Proceedings of the Fifth Annual International Conference on Computational Biology, pp. 143–202. ACM Press, New York (2001)
20. Menke, M., Berger, B., Cowen, L.: Matt: local flexibility aids protein multiple structure alignment. PLoS Comput. Biol. 4(1), e10 (2008)
21. Oakley, M.T., Barthel, D., Bykov, Y., Garibaldi, J.M., Burke, E.K., Krasnogor, N., Hirst, J.D.: Search strategies in structural bioinformatics. Curr. Protein Pept. Sci. 9(3), 260–274 (2008)
22. Strickland, D.M., Barnes, E., Sokol, J.S.: Optimal protein structure alignment using maximum cliques. Oper. Res. 53(3), 389–402 (2005)

23. Subbiah, S., Laurents, D.V., Levitt, M.: Structural similarity of dna-binding domains of bacteriophage repressors and the globin core. Curr. Biol. 3(3), 141–148 (1993)
24. Wang, Y., Makedon, F., Ford, J., Huang, H.: A bipartite graph matching framework for finding correspondences between structural elements in two proteins. In: Engineering in Medicine and Biology Society, IEMBS 2004. 26th Annual International Conference of the IEEE, September 2004, vol. 2, pp. 2972–2975 (2004)
25. Xie, W., Sahinidis, N.V.: A reduction-based exact algorithm for the contact map overlap problem. J. Comput. Biol. 14(5), 637–654 (2007)

Predicting and Analyzing Cellular Networks
(Invited Keynote Talk)

Mona Singh

Lewis-Sigler Institute for Integrative Genomics
Department of Computer Science
Princeton University, Princeton NJ 08544
mona@cs.princeton.edu

High-throughput experimental technologies, along with computational predictions, have resulted in large-scale biological networks for numerous organisms. Global analyses of biological networks provide new opportunities for revealing protein functions and pathways, and for uncovering cellular organization principles. In my talk, I will discuss a number of approaches we have developed over the years for the complementary problems of predicting interactions and analyzing interaction networks. First, I will describe a genomic approach for uncovering high-confidence regulatory interactions, and show how it can be effectively combined with a framework for predicting regulatory interactions for proteins with known structural domains but unknown binding specificity. Next, I will describe algorithms for analyzing protein interaction networks in order to uncover protein function and functional modules, and demonstrate the importance of considering the topological structure of interaction networks in order to make high quality predictions. Finally, I will present a framework for explicitly incorporating known attributes of individual proteins into the analysis of biological networks, and utilize it to discover recurring network patterns underlying a range of biological processes.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, p. 166, 2010.

A Consensus Tree Approach for Reconstructing Human Evolutionary History and Detecting Population Substructure

Ming-Chi Tsai[1], Guy Blelloch[2], R. Ravi[3], and Russell Schwartz[4]

[1] Joint CMU-Pitt Computational Biology Program,
Carnegie Mellon University and University of Pittsburgh, Pittsburgh, PA 15213, USA
[2] Department of Computer Science
[3] Tepper School of Business
[4] Department of Biological Science,
Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract. The random accumulation of variations in the human genome over time implicitly encodes a history of how human populations have arisen, dispersed, and intermixed since we emerged as a species. Reconstructing that history is a challenging computational and statistical problem but has important applications both to basic research and to the discovery of genotype-phenotype correlations. In this study, we present a novel approach to inferring human evolutionary history from genetic variation data. Our approach uses the idea of consensus trees, a technique generally used to reconcile species trees from divergent gene trees, adapting it to the problem of finding the robust relationships within a set of intraspecies phylogenies derived from local regions of the genome. We assess the quality of the method on two large-scale genetic variation data sets: the HapMap Phase II and the Human Genome Diversity Project. Qualitative comparison to a consensus model of the evolution of modern human population groups shows that our inferences closely match our best current understanding of human evolutionary history. A further comparison with results of a leading method for the simpler problem of population substructure assignment verifies that our method provides comparable accuracy in identifying meaningful population subgroups in addition to inferring the relationships among them.

1 Introduction

The advent of high-throughput genotyping methods and their application in large-scale genetic variation studies have made it possible to determine in unprecedented detail how the modern diversity of the human species arose from our common ancestors. In addition to its importance as a basic research problem, this topic has great practical relevance to the discovery of genetic risk factors of disease due to the confounding effect of unrecognized substructure on genetic association tests [22]. Past work on human ancestry inference has essentially treated it as two distinct problems: identifying meaningful population groups

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 167–178, 2010.

and inferring evolutionary trees among them. Population groups may be assumed in advance based on common conceptions of ethnic groupings, although the field increasingly depends on computational analysis to make such inferences automatically. Probably the most well known system for this problem is STRUCTURE [16], which uses a Markov Chain Monte Carlo (MCMC) clustering method to group sequences into subpopulations characterized by similar allele frequencies across variation sites. A variety of other computational and statistical methods have been developed to perform population substructure inference or similar analyses, including EIGENSOFT [15], Spectrum [19], and SABER [21]. A separate literature has arisen on the inference of relationships between populations, typically based on phylogenetic reconstruction of limited sets of genetic markers — such as classic restriction fragment length polymorphisms [14], mtDNA genotypes [9,2], short tandem repeats [9,23], and Y chromosome polymorphism [5] — supplemented by extensive manual analysis informed by population genetics theory. There has thus far been little cross-talk between the two problems of inferring population substructure and inferring phylogenetics of subgroups, despite the fact that both problems depend on similar data sources and in principle can help inform the decisions of one another.

We propose a novel approach for reconstructing a species history that is intended to unify these two inference problems. The method is conceptually based on the idea of consensus trees [13], which represent inferences as to the robust features of a family of trees. The approach takes advantage of the fact that the availability of large-scale variation data sets, combined with new algorithms for fast phylogeny inference on these data sets [20], has made it possible to infer likely phylogenies on millions of small regions spanning the human genome. The intuition behind our method is that each such phylogeny will represent a distorted version of the global evolutionary history and population structure of the species, with many trees supporting the major splits or subdivisions between population groups while few support any particular splits independent of those groups. By detecting precisely the robust features of these trees, we can assemble a model of the true evolutionary history and population structure that can be made resistant to overfitting and to noise in the SNP data or tree inferences.

In the remainder of this paper, we describe and evaluate our approach. We first present in more detail our mathematical model of the consensus tree problem and a set of algorithms for finding consensus trees from families of local phylogenies. We next evaluate our method on the HapMap Phase II [7] and Human Genome Diversity Project [8] datasets. Finally, we consider some of the implications of the results and future prospects of the consensus tree approach for evolutionary history and substructure inference.

2 Methods

2.1 Consensus Tree Model

We assume we are given a set of m taxa, S, representing the paired haplotypes from each individual in a population sample. If we let $\mathcal{T}$ be the set of all possible

labeled trees connecting the $s \in S$, where each node of any $t \in T$ may be labeled by any subset of zero or more $s \in S$ without repetition, then our input will consist of some set of n trees $\mathcal{D} = (T_1, \ldots, T_n) \subseteq \mathcal{T}$. Our desired output will also be some labeled tree $T_M \in \mathcal{T}$, intended to represent a consensus of $T_1, \ldots, T_n$.

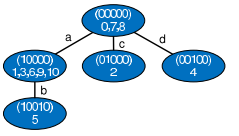
Our objective function for choosing T_M is based on the task of finding a consensus tree [13] from a set of phylogenies each describing inferred ancestry of a small region of a genome. Our problem is, however, fairly different from standard uses of consensus tree algorithms in that our phylogenies are derived from many variant markers, each only minimally informative, within a single species. Standard consensus tree approaches, such as majority consensus [11] or Adam consensus [1], would not be expected to be effective in this situation as it is likely there is no single subdivision of a population that is consistently preserved across more than a small fraction of the local intraspecies trees and that many similar but incompatible subdivisions are supported by different subsets of the trees. We therefore require an alternative representation of the consensus tree problem designed to be robust to large numbers of trees and high levels of noise and uncertainty in data.

For this purpose, we chose a model of the problem based on the principle of minimum description length (MDL)[4], a standard technique for avoiding overfitting when making inferences from noisy data sets. An MDL method seeks to minimize the amount of information needed to encode the model and to encode the data set given knowledge of the model. Suppose we have some function $L : \mathcal{T} \rightarrow \mathcal{R}$ that computes a description length, $L(T_i)$, for any tree T_i. We will assume the existence of another function, which for notational convenience we will also call L, $L : \mathcal{T} \times \mathcal{T} \rightarrow \mathcal{R}$, which computes a description length, $L(T_i|T_j)$, of a tree T_i given that we have reference to a model T_j. Then, given a set of observed trees, $\mathcal{D} = \{T_1, T_2, ..., T_n\}$ for $T_i \in \mathcal{T}$, our objective function is

$$\mathcal{L}(T_M, T_1, \ldots, T_n) = \arg \min_{T_M \in \mathcal{T}} \left(L(T_M) + \sum_{i=1}^{n} L(T_i|T_M) + f(T_M) \right)$$

The first term computes the description length of the model (consensus) tree T_M. The sum computes the cost of explaining the set of observed (input) trees $\mathcal{D}$. The function $f(T_M) = |T_M| \log_2 m$ defines an additional penalty on model edges used to set a minimum confidence level on edge predictions.

We next need to specify how we compute the description length of a tree. For this purpose, we use the fact that a phylogeny can be encoded as a set of bipartitions (or *splits*) of the taxa with which it is labeled, each specifying the set of taxa lying on either side of a single edge of the tree. We represent the observed trees and candidate consensus trees as sets of bipartitions for the purpose of calculating description lengths. Once we have identified a set of bipartitions representing the desired consensus tree, we then apply a tree reconstruction algorithm to convert those bipartitions into a tree. A bipartition b can in turn be represented as a string of bits by arbitrarily assigning elements in one part of the bipartition the label "0" and the other part the label "1". Fig. 1a shows an example of a hypothetical tree, its description as a set of bipartitions, and



(a)

e_a: 1,3,5,6,9,10|0,2,4,7,8
e_b: 0,1,2,3,4,6,7,8,9,10|5
e_c: 0,1,3,4,5,6,7,8,9,10|2
e_d: 0,1,2,3,5,6,7,8,9,10|4

(b)

e_a: 01010110011
e_b: 00000100000
e_c: 00100000000
e_d: 00001000000

(c)

Fig. 1. (a) A maximum parsimony (MP) tree consisting of 11 labeled individuals or haplotypes. (b) The set of bipartitions induced by edges (e_a, e_b, e_c, e_d) in the tree. (c) 0-1 bit sequence representation for each bipartition.

representations of the bipartitions as bit strings. Such a bit representation allows us to compute the encoding length of a bipartition b as the entropy of its corresponding bit string. If we define p_0 to be the fraction of bits of b that are zero and p_1 as the fraction that are one, then:

$$L(b) = m\left(-p_0 \log_2 p_0 - p_1 \log_2 p_1\right)$$

Similarly, we can encode the representation of one bipartition b_1 given another b_2 using the concept of conditional entropy. If we let p_{00} be the fraction of bits for which both bipartitions have value "0," p_{01} be the fraction for which the first bipartition has value "0" and the second "1," and so forth, then:

$$L(b_1|b_2) = m\left[\sum_{s,t\in\{0,1\}} -p_{st} \log_2 p_{st} \; + \sum_{u\in\{0,1\}} (p_{0u} + p_{1u}) \log_2 (p_{0u} + p_{1u})\right]$$

where the first term is the joint entropy of b_1 and b_2 and the second term is the entropy of b_2.

We can use these definitions to specify the minimum encoding cost of a tree $L(T_i)$ or of one tree given another $L(T_i|T_M)$. We first convert the tree into a set of bipartitions $b_1, \ldots, b_k$. We can then observe that each bipartition b_i can be encoded either as an entity to itself, with cost equal to its own entropy $L(b_i)$, or by reference to some other bipartition b_j with cost $L(b_i|b_j)$. In addition, we must add a cost for specifying whether each b_i is explained by reference to another bipartition and, if so, which one. The total minimum encoding costs, $L(T_M)$ and $L(T_i|T_M)$, can then computed by summing the minimum encoding cost for each bipartition in the tree. Specifically, let $b_{t,i}$ and $b_{s,M}$ be elements from the bipartition set B_i of T_i and B_M of T_M, respectively. We can then compute $L(T_M)$ and $L(T_i|T_M)$ by optimizing for the following objectives over possible reference bipartitions, if any, for each bipartition in each tree:

$$L(T_M) = \underset{b_s \in B_M \cup \{\emptyset\}}{\arg\min} \sum_{s=1}^{|B_M|} \left[L(b_{s,M}|b_s) + \log_2\left(|B_M| + 1\right)\right]$$

$$L(T_i|T_M) = \underset{b_t \in B_M \cup B_i \cup \{\emptyset\}}{\arg\min} \sum_{t=1}^{|B_i|} \left[L(b_{t,i}|b_t) + \log_2\left(|B_M| + |B_i| + 1\right)\right]$$

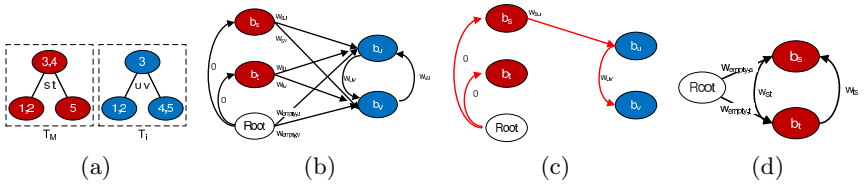


Fig. 2. Illustration of the DMST construction for determining model description length. (a) Hypothetical model tree T_M (red) and observed tree T_i (blue). (b) Graph of possible reference relationships for explaining T_i (blue nodes) by reference to T_M (red nodes). (c) A possible resolution of the graph of (b). (d) Graph of possible reference relationships for explaining T_M by itself.

2.2 Algorithms

Encoding Algorithm. We pose the problem of computing $L(T_M)$ and $L(T_i|T_M)$ as a weighted directed minimum spanning tree (DMST) problem, illustrated in Fig. 2. We construct a graph $G = (V, E)$ in which each node represents either a bipartition or a single "empty" root node r explained below. Each directed edge (b_j, b_i) represents a possible reference relationship by which b_j explains b_i. If a bipartition b_i is to be encoded from another bipartition b_j, the weight of the edge e_{ji} would be given by $w_{ji} = L(b_i|b_j) + \log_2 |V|$ where the term $\log_2 |V|$ represents the bits we need to specify the reference bipartition (including no bipartition) from which b_i might be chosen. This term introduces a penalty to avoid overfitting. We add an additional edge directly from the empty node to each node to be encoded whose weight is the cost of encoding the edge with reference to no other edge, $w_{empty,j} = L(b_j) + \log_2 |V|$.

To compute $L(T_M)$, the bipartitions B_M of T_M and the single root node collectively specify the complete node set of the directed graph. One edge is then created from every node $B_M \cup \{r\}$ to every node of B_M. To compute $L(T_i|T_M)$, the node set will include the bipartitions B_i of T_i, the bipartitions B_M of T_M, and the root node r. The edge set will consist of two parts. Part one consists of one edge from each node of $B_i \cup B_M \cup \{r\}$ to each node of B_i, with weights corresponding to the cost of possible encodings of B_i. Part two will consist of a zero-cost edge from r to each node in B_M, representing the fact that the presumed cost of the model tree has already been computed. Fig. 2 illustrates the construction for a hypothetical model tree T_M and observed tree T_i (Fig. 2(a)), showing the graph of possible reference relationships (Fig. 2(b)), a possible solution corresponding to a specific explanation of T_i in terms of T_M (Fig. 2(c)), and the graph of possible reference relationships for T_M by itself (Fig. 2(d)).

For both constructions, the minimum encoding length is found by solving for the DMST with the algorithm of Chiu and Liu [3] and summing the weights of the edges. This cost is computed for a candidate model tree T_M and for each observed tree T_i to give the total cost $[\mathcal{L}(T_M, T_1, \ldots, T_n)]$.

Tree Search. While the preceding algorithm gives us a way to compute the score of any possible consensus tree T_M, we still require a means of finding a high-quality (low-scoring) tree. The space of possible trees is too large to permit exhaustive search and we are unaware of an efficient algorithm for finding a global optimum of our objective function. We therefore employ a heuristic search strategy based on simulated annealing. The algorithm relies on the intuition that the bipartitions to be found in any high-quality consensus tree are likely to be the same as or similar to bipartitions frequently observed in the input trees. The algorithm runs for a total of t iterations and at each iteration i will either insert a new bipartition chosen uniformly at random from the observed (non-unique) bipartitions with probability $1 - i/t$ or delete an existing bipartition chosen uniformly at random from the current T_M with probability i/t to create a candidate model tree T'_M. If the algorithm chooses to insert a new bipartition b, it then performs an additional expectation-maximization-like local optimization to improve the fit. It repeatedly identifies the set B of bipartitions explained by b and then locally improves b by iteratively flipping any bits that lower the cost of explaining B, continuing until it converges on some locally optimal b. This final bipartition is then added to T_M to yield the new candidate tree T'_M. Once a new candidate tree T'_M has been established, the algorithm tests the difference in cost between T_M and T'_M. If T'_M has reduced cost then the move is accepted and T'_M becomes the new starting tree. Otherwise, the method accepts T'_M with probability $p = \exp \frac{\mathcal{L}(T_M,T_1,\ldots,T_n)-\mathcal{L}(T'_M,T_1,\ldots,T_n)}{T}$ where $T = 400/t$ is the simulated annealing temperature parameter.

Tree Reconstruction. A final step in the algorithm is the reconstruction of the consensus tree from its bipartitions. We first sort the model bipartitions $b_1 \prec b_2 \ldots \prec b_k$ in decreasing order of numbers of splits they explain (i.e., the number of out-edges from their corresponding nodes in the DMST). We then initialize a tree T_0 with a single node containing all haplotype sequences in S and introduce the successive bipartitions in sorted order into this tree. For each $b_i = 1$ to k, we subdivide any node v_j that contains elements with label 0 in b_i (b_i^0) and elements labeled as 1 in b_i (b_i^1) into nodes v_{j1} and v_{j2} corresponding to the subpopulations of v_j in b_i^0 or b_i^1. We also introduce a Steiner node s_j for each node v_j to represent the ancestral population from which v_{j1} and v_{j2} diverged. We then replace the prior tree T_{i-1} with $T_i = (V_i, E_i)$ where $V_i = V_{i-1} - \{v_j\} + \{v_{j1}, v_{j2}, s_j\}$ and $E_i = E_{i-1} - \{e = (t, v_j) | e \in E_{i-1}, t \in \text{parent}(v_j)\} + \{e = (t, s_j) | t \in \text{parent}(v_j)\} + \{(s_j, v_{j1}), (s_j, v_{j2})\}$. After introducing all k bipartitions, T_k is then the final consensus tree. The number of bipartitions w_j explained by each model bipartition b_j provides a rough estimate of the number of mutations that occurred after the population diverged, which can be interpreted as an estimated elapsed time scaled by population size. We attribute this scaled time equally to the two branches to assign branch lengths to the tree. Given a weight w_j for the j-th model bipartition, the branch length of $e = (s_j, v_{j1})$ and (s_j, v_{j2}) would then be $w_j/2$ and the branch length of $e = (t, s_j)$ for $t = \text{parent}(v_j)$ would be $w_{j-1}/2 - w_j/2$.

2.3 Validation Experiments

We evaulated our methods by applying them to samples from two SNP variation datasets. We first used the phase II HapMap data set (phased, release 22) [7] which consists of over 3.1 million SNP sites genotyped for 270 individuals from four populations: 90 Utah residents with ancestry from Northern and Western Europe (CEU); 90 individuals with African ancestry from Ibadan, Nigeria (YRI); 45 Han Chinese from Beijing, China (CHB); and 44 Japanese in Tokyo, Japan (JPT). For the CEU and YRI groups, which consist of trio data (parents and a child), we used only the 60 unrelated parents with genotypes as inferred by the HapMap consortium. For each run, we randomly sampled 8,000 trees each constructed from 5 consecutive SNPs uniformly at random from 45,092 trees generated from chromosome 21, which represented an average of 28,080 unique SNPs. For the purpose of comparison, we used 8,000 trees or the corresponding 28,080 SNPs as inputs to our method and the comparative algorithms. We next used phased data (version 1.3) from the Human Genome Diversity Project (HGDP) [8], which genotyped 525,910 SNP sites in 597 individuals from 29 populations categorized into seven region of origin: Central South Asia (50 individuals), Africa (159 individuals), Oceania (33 individuals), Middle East (146 individuals), America (31 individuals), East Asia (90 individuals), and Europe (88 individuals). For each test with the HGDP data, we sampled 10,000 trees from a set of 39,654 trees uniformly at random from chromosome 1. The 10,000 trees on average consisted of 30,419 unique SNPs.

We are not aware of any comparable method to ours and therefore cannot directly benchmark it against any competitor. We therefore assessed it by two criteria. We first assessed the quality of the inferred population histories by reference to a expert-curated model of human evolution derived from a review by Shriver and Kittles[18], which we treat as a "gold standard." Shriver and Kittles used a defined set of known human population groups rather than the coarser grouping inferred by our method. To allow comparison with either of our inferred trees, we therefore merged any subgroups that were joined in our tree but distinct in the Shriver tree and deleted any subgroups corresponding to populations not represented in the samples from which our trees were inferred. (For example, for the HapMap Phase II dataset, we removed Melanesian, Polynesian, Middle Eastern, American, and Central South Asian subgroups from the tree, as individuals from those populations were not typed in the Phase II HapMap). We also ignored inferred admixture events in the Shriver and Kittles tree. We then manually compared our tree to the resulting condensed version of the Shriver and Kittles "gold standard" tree.

As a secondary validation, we also assessed the quality of our inferred population subgroups relative to those inferred by one of the leading substructure algorithms, STRUCTURE (version 2.2) [16]. STRUCTURE requires that the user specify a desired number of populations, for which we supplied the true number for each data set (four for HapMap and seven for HGDP). For each run, we performed 2,000 iterations of burn-ins and 10,000 iterations of the STRUCTURE MCMC sampling, assigning each individual to the population group of

highest likelihood as determined by STRUCTURE. We did not make use of STRUCTURE's capacity to infer admixture or to use additional data on linkage disequilibrium between sites. We assessed the quality of the results based on variation of information [12], a method commonly used to assess accuracy of a clustering method relative to a pre-defined "ground truth." Variation of information is defined as $2H(X,Y) - H(X) - H(Y)$, where $H(X,Y)$ is the joint entropy of the two labels (inferred clustering and ground truth) and $H(X)$ and $H(Y)$ are their individual entropies. We also assessed robustness of the methods to repeated subsamples. For each pair of individuals (i,j) across five independent samples, we computed the number of samples a_{ij} in which those individuals were grouped in the same cluster and the number b_{ij} in which they were grouped in different clusters. Each method was assigned an overall inconsistency score of $\sum_{i,j} \min\{1 - \frac{2b_{ij}}{\lfloor (a_{ij}+b_{ij})\rfloor}, 1 - \frac{2a_{ij}}{\lfloor (a_{ij}+b_{ij})\rfloor}\} / \binom{n}{2}$. The measure will be zero if clusters are perfectly consistent from run-to-run and approach one for completely inconsistent clustering. We defined the ground truth for HapMap as the four population groups. For the HGDP data, we treated the ground truth as the seven regions of origin rather than the 29 populations, because many population groups are genetically similar and cannot be distinguished with limited numbers of SNPs.

3 Results

Fig. 3 shows the trees inferred by our method on the two data sets alongside their corresponding condensed Shriver and Kittles "gold standard" trees. Fig. 3(a) shows the inferred tree produced by our model. Based on the numbers of bipartitions explained by each method, the tree reconstruction infers there to be an initial separation of the YRI (African) sub-population from the others (CEU+JPT+CHB) followed by a subsequent separation of CEU (European) from JPT+CHB (East Asian). When collapsed to the same three populations (African, European, East Asian), the gold standard tree (Fig. 3(b)) shows an identical structure. Furthermore, these results are consistent with many independent lines of evidence for the out-of-Africa hypothesis of human origins [10,24,18].

For the HGDP dataset, the trees differ slightly from run to run, so we arbitrarily provide our first run, Fig. 3(c), as a representative. The tree infers the most ancient divergence to be that between Africans and the rest of the population groups, followed by a separation of Oceanian from other non-Africans, a separation of Asian+American from European+Middle Eastern (and a subset of Central South Asian), and then a more recent split of American from Asian. Finally, a small cluster of just two Middle Eastern individuals is inferred to have separated recently from the rest of the Middle Eastern, European, and subset of Central South Asian. The tree is nearly identical to the that derived from Shriver and Kittles for the same population groups (Fig. 3(d)). The only notable distinctions are that gold standard tree has no equivalent to our purely Middle Eastern node; that the gold standard does not distinguish between the divergence times of Oceanian and other non-African populations from the African

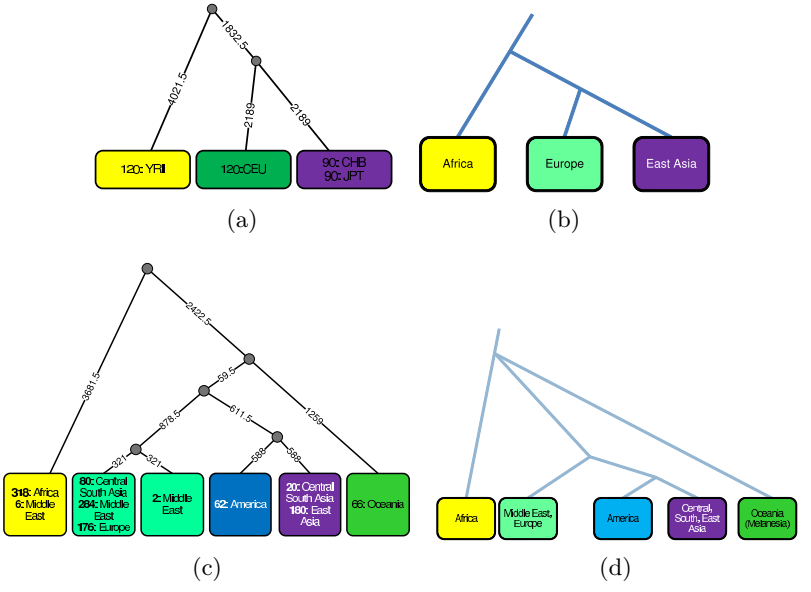
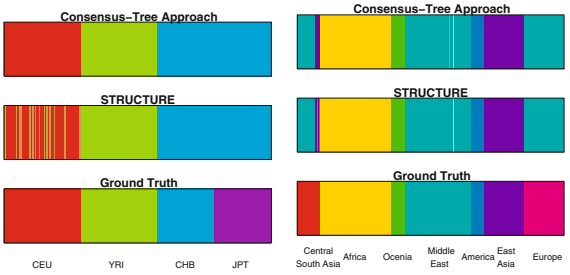


Fig. 3. Inferred consensus trees. Node labels show numbers of haplotypes belonging to each known populations. Edge labels can be interpreted as estimates of scaled time since each divergence. (a) Consensus tree obtained from HapMap dataset. (b) Trimmed and condensed tree from [18]. (c) Consensus tree obtained from HGDP dataset. (d) Trimmed and condensed tree from [18].

while ours predicts a divergence of Oceanian and European/Asian well after the African/non-African split; and that the gold standard groups Central South Asian with East Asians while ours splits Central South Asian groups between European and East Asian subgroups (an interpretation supported by more recent analyses [17]). Our results are also consistent with the simpler picture provided by the HapMap data as well as with a general consensus in the field derived from many independent phylogenetic analyses [25,10].

Fig. 4 shows the corresponding cluster assignments for our method and STRUCTURE in order to provide a secondary assessment of our method's utility for the simpler sub-problem of subpopulation inference relative to STRUCTURE and the presumed ground truth. Each inferred cluster is assigned a distinct label, with colors chosen to maximize agreement with the true population structure. For HapMap (Fig. 4(a)), our method consistently identified YRI and CEU as distinct subpopulations but failed to separate CHB (Chinese) and JPT (Japanese). STRUCTURE produced generally identical output except in one run where it grouped a subset of the CHB and JPT populations in a separate cluster. Tab. 1(a) quantifies these observations, suggesting marginally better performance for the consensus tree method by both measures. Results were more ambiguous for HGDP (Fig. 4(b)) with STRUCTURE showing generally greater sensitivity but still worse consistency than our method. STRUCTURE usually at least approximately finds six of the annotated seven population groups, having



(a) (b)

Fig. 4. Inferred population structures from the consensus tree method and STRUCTURE. From top to bottom: consensus-tree, STRUCTURE, and ground truth. (a): Inferred population structures from a single trial of 8,000 trees from HapMap Phase II dataset. (b): Inferred population structures from one trial of 10,000 trees.

Table 1. Variation of information (VI) and inconsistency score. Lower VI reflects higher accuracy in identifying known population structure. Higher consistency reflects greater reproducibility between independent samples.

(a) Hapmap

	VI	Consistency
STRUCTURE	0.5039	0.0226
Consensus Tree	**0.4286**	**0.0000**

(b) HGDP

	VI	Consistency
STRUCTURE	**0.8949**	0.1341
Consensus Tree	0.9265	**0.0765**

difficulty only in identifying Central South Asians as a distinct group, consistent with a similar outcome from He *et al.* [6]. The consensus tree method reliably finds five of the seven populations, usually conflating Middle Eastern and European in addition to failing to recognize Central South Asians. Tab. 1(b) quantifies these observations, with the consensus tree method showing slightly worse variation of information but better consistency than STRUCTURE. We note that our methods also provide comparable runtimes to STRUCTURE despite solving a more involved inference problem. Our methods required approximately 1.4 hours for the HapMap data and 30 hours for the HGDP data, compared to approximately 2.5 hours and 48 hours for STRUCTURE.

4 Discussion

We have presented a novel method for simultaneously inferring population ancestries and identifying population subgroups. The method builds on the general concept of a "consensus tree" summarizing the output of many independent sources of information, using a novel MDL realization of the consensus tree concept to allow it to make robust inferences across large numbers of measurements, each individually minimally informative. It incidentally provides a *de novo* inference of population subgroups comparable in quality to that provided by the

leading STRUCTURE method. The method also provides edge length estimates that can roughly be interpreted as estimates of time since divergence on the crude assumption that effective population sizes are equal along all sibling tree edges. The addition of an outgroup to determine likely ancestral states at internal nodes of the tree should in principle allow us to drop that assumption and estimate both divergence times and effective population sizes along the tree edges. The MDL approach should also in principle automatically adapt to larger data sets, producing more detailed inferences as the data to support them becomes available. In future work, we hope to better test these assumptions, in part by developing protocols for simulating sequence data generated from a human-like population history, and to extend the method to inferences of ancestry in the presence of admixture.

Acknowledgments

The authors would like to thank Srinath Sridhar for valuable discussions on the ideas behind this work. This work was supported by U.S. National Science Foundation IIS award #0612099 and by NIH T32 training grant T32 EB009403 as part of the HHMI-NIBIB Interfaces Initiative.

References

1. Adams, E.N.: N-trees as nestings: Complexity, similarity, and consensus. Journal of Classification 3(2), 299–317 (1986) 10.1007/BF01894192
2. Cann, R.L., Stoneking, M., Wilson, A.C.: Mitochondrial DNA and human evolution. Nature 325(6099), 31–36 (1987) 10.1038/325031a0
3. Chu, Y.J., Liu, T.H.: On the shortest arborescence of a directed graph. Science Sinica 14, 1396–1400 (1965)
4. Grnwald, P.D., Myung, I.J., Pitt, M.A.: Advances in Minimum Description Length: Theory and Applications. The MIT Press, Cambridge (2005)
5. Hammer, M.F., Spurdle, A.B., Karafet, T., Bonner, M.R., Wood, E.T., Novelletto, A., Malaspina, P., Mitchell, R.J., Horai, S., Jenkins, T., Zegura, S.L.: The geographic distribution of human Y chromosome variation. Genetics 145(3), 787–805 (1997)
6. He, M., Gitschier, J., Zerjal, T., de Knijff, P., Tyler-Smith, C., Xue, Y.: Geographical affinities of the HapMap samples. PLoS ONE 4(3), e4684, 03 (2009)
7. International HapMap Consortium. A second generation human haplotype map of over 3.1 million snps. Nature 449(7164), 851–861 (October 2007)
8. Jakobsson, M., Scholz, S.W., Scheet, P., Gibbs, R.J., Vanliere, J.M., Fung, H.C., Szpiech, Z.A., Degnan, J.H., Wang, K., Guerreiro, R., Bras, J.M., Schymick, J.C., Hernandez, D.G., Traynor, B.J., Simon-Sanchez, J., Matarin, M., Britton, A., van de Leemput, J., Rafferty, I., Bucan, M., Cann, H.M., Hardy, J.A., Rosenberg, N.A., Singleton, A.B.: Genotype, haplotype and copy-number variation in worldwide human populations. Nature 451(7181), 998–1003 (2008)
9. Jorde, L.B., Bamshad, M.J., Watkins, W.S., Zenger, R., Fraley, A.E., Krakowiak, P.A., Carpenter, K.D., Soodyall, H., Jenkins, T., Rogers, A.R.: Origins and affinities of modern humans: a comparison of mitochondrial and nuclear genetic data. American Journal of Human Genetics 57, 523–538 (1995)

10. Kayser, M., Krawczak, M., Excoffier, L., Dieltjes, P., Corach, D., Pascali, V., Gehrig, C., Bernini, L.F., Jespersen, J., Bakker, E., Roewer, L., de Knijff, P.: An extensive analysis of Y-chromosomal microsatellite haplotypes in globally dispersed human populations. American Journal of Human Genetics 68(4), 990–1018 (2001)
11. Margush, T., Mcmorris, F.R.: Consensus n-trees. Bulletin of Mathematical Biology 43, 239–244 (1981)
12. Meila, M.: Comparing clusterings–an information based distance. Journal of Multivariate Analysis 98(5), 873–895 (2007) doi: 10.1016/j.jmva.2006.11.013
13. Nei, M., Kumar, S.: Molecular Evolution and Phylogenetics. Oxford University Press, Oxford (2000)
14. Nei, M., Roychoudhury, A.K.: Genetic relationship and evolution of human races. Evolutionary Biology 14, 1–59 (1982)
15. Patterson, N., Price, A.L., Reich, D.: Population structure and eigenanalysis. PLoS Genetics 2(12), e190+ (2006)
16. Pritchard, J.K., Stephens, M., Donnelly, P.: Inference of population structure using multilocus genotype data. Genetics 155(2), 945–959 (2000)
17. Reich, D., Thangaraj, K., Patterson, N., Price, A.L., Singh, L.: Reconstructing indian population history. Nature 461(7263), 489–494 (2009) 10.1038/nature08365
18. Shriver, M.D., Kittles, R.A.: Genetic ancestry and the search for personalized genetic histories. Nature Reviews Genetics 5, 611–618 (2004)
19. Sohn, K.A., Xing, E.P.: Spectrum: joint bayesian inference of population structure and recombination events. Bioinformatics 23(13), i479–i489 (2007)
20. Sridhar, S., Lam, F., Blelloch, G., Ravi, R., Schwartz, R.: Direct maximum parsimony phylogeny reconstruction from genotype data. BMC Bioinformatics 8(1), 472 (2007)
21. Tang, H., Coram, M., Wang, P., Zhu, X., Risch, N.: Reconstructing genetic ancestry blocks in admixed individuals. The American Journal of Human Genetics 79(1), 1–12 (2006) doi: 10.1086/504302
22. Thomas, D.C., Witte, J.S.: Point: Population stratification: A problem for case-control studies of candidate-gene associations? Cancer Epidemiol Biomarkers Prev. 11(6), 505–512 (2002)
23. Tishkoff, S.A., Dietzsch, E., Speed, W., Pakstis, A.J., Kidd, J.R., Cheung, K., Bonn-Tamir, B., Santachiara-Benerecetti, A.S., Moral, P., Krings, M., Pbo, S., Watson, E., Risch, N., Jenkins, T., Kidd, K.K.: Global patterns of linkage disequilibrium at the CD4 locus and modern human origins. Science 271(5254), 1380–1387 (1996)
24. Tishkoff, S.A., Verrelli, B.C.: Patterns of human genetic diversity: Implications for human evolutionary history and disease. Annual Review of Genomics and Human Genetics 4(1), 293–340 (2003)
25. Tishkoff, S.A., Williams, S.M.: Genetic analysis of African populations: human evolution and complex disease. Nat. Rev. Genet. 3(8), 611–621 (2002) 10.1038/nrg865

Inferring Evolutionary Scenarios for Protein Domain Compositions

John Wiedenhoeft[1], Roland Krause[1,2], and Oliver Eulenstein[3]

[1] Free University of Berlin
john.wiedenhoeft@fu-berlin.de
[2] Max Planck Institute for Molecular Genetics Berlin
roland.krause@molgen.mpg.de
[3] Iowa State University
oeulenst@cs.iastate.edu

Abstract. Essential cellular processes are controlled by functional interactions of protein domains, which can be inferred from their evolutionary histories. Methods to reconstruct these histories are challenged by the complexity of reconstructing macroevolutionary events. In this work we model these events using a novel network-like structure that represents the evolution of domain combinations, called plexus. We describe an algorithm to find a plexus that represents the evolution of a given collection of domain histories as phylogenetic trees with the minimum number of macroevolutionary events, and demonstrate its effectiveness in practice.

1 Introduction

Inferring the evolutionary history of domain compositions of proteins is a key problem for the elucidation of protein function from large-scale genomic data. In essence, a *domain* is an independent and evolutionary mobile sub-unit of a protein [1]. The recognition of such characteristics has led to breakthroughs in the determination of protein function, e. g. for the oncogene BRCA1 [2]. The vast majority of proteins in the higher Eukaryotes consist of several domains [3]. About 200 of these domains combine frequently into a rich variety of *multi-domain proteins* (MDPs) that are involved in essential cellular processes, including chromatin remodeling and signal transduction [4]. Recombination events of domains lead to similarities between proteins that have more than one common ancestor, and which are therefore not strictly homologous. These proteins can pose a major problem for phylogenetic inference in protein families [5].

Here, we describe a novel approach to reconstruct evolutionary MDP scenarios for which standard phylogenetic inference methods may not be appropriate. We formulate the *MDP evolution problem*, describe an effective heuristic to solve it and show that its implementation performs well in practice.

1.1 Background

After the discovery of mobile domain combinations in the 1980s, it required complete eukaryotic genome sequences for thorough investigations of the

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 179–190, 2010.

phenomenon [1]. Genome wide studies of multi-domain proteins either utilize the order of the domains or study the co-occurence, but typically ignore relationships of the sequence fragments and do not attempt to map individual macro-evolutionary events. Quantitative studies found the number of observed neighbors for a domain to follow a power-law distribution [6].

Phylogeny-oriented work concentrated on analyzing evolutionary events that establish multiple-domain compositions, and derive phylogenetic trees from domain combinations using parsimony-based criteria or clustering approaches [7,8]. [9] used a parsimony-based approach and simplified gene fusion, domain shuffling and retrotransposition events into tractable merge and deletion operations. [10] constructed a more elaborate model with 3 subclasses of fusion events for multi-domain proteins to reconstruct domain trees.

Previous work mostly investigated general principles of protein evolution. In contrast, methods for the reconstruction of MDP histories based on macro-evolutionary events are still in their infancy, and studies of particular protein families typically resorted to manual annotation [11,12].

[13] suggested an approach incorporating domain histories to reconstruct ancestral domain compositions from a given collection of domain trees and a given species tree. Each domain-node of a domain tree is mapped to a node of the species tree. The domains in a species node are partitioned to represent multi-domain proteins in the parent species with the weighted minimum number of merges and deletions in comparison to the child species. The method relies on the following critical assumptions: the correctness of the domain trees, the correctness of a species tree, and the correct mapping of each domain-node into the species tree, all of which may not be satisfiable in practice.

Suitably restricted networks to model macro-evolution events have been explored where trees are no longer sufficient and several interesting approaches were used with success for phylogenetic displays and mapping of events, reviewed in [14]. Our approach relates to [15], which is aimed at the reconstruction of phylogenies with recombination events. However, this and similar models are not directly applicable to reconstruct the evolution of MDPs.

1.2 Contribution of This Work

Our formalization of the *MDP evolution problem* is: given a collection of phylogenetic trees of extant domains, find scenarios that minimize the change in MDP composition. We describe an effective heuristic for this reconstruction problem and show that its implementation performs well in practice for a selection of proteins with frequently recombining domains. We do not rely on a given species tree but present a novel graph-theoretic network, called *plexus*, that allows to describe scenarios for the evolution of a collection of domain trees. We introduce three different instances of this network (see Fig. 1). The *expanded* plexus corresponds to a biological scenario, the *reconstructable* to what is obtainable from the phylogenetic reconstruction and the *compact* to a computationally feasible model.

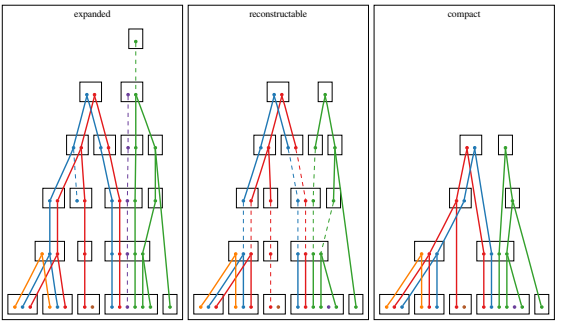


Fig. 1. Counterparts of plexūs for a set of MDPs and their domain trees. Edges mark inheritance relations between domains (dots) in MDPs (rectangles), different domain families are in different shades. The expanded plexus consists of evolutionary events only (see Fig. 2). Its non-reconstructable edges are dashed and disappear in the reconstructable counterpart. Some of the resulting blocks may then contain only nodes with out-degree 1 (dashed). Contracting their out-edges results in the compact counterpart.

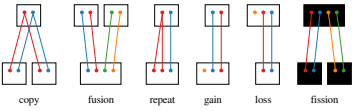


Fig. 2. Basic events in MDP evolution. Fission is modeled with elementary events (see Fig. 3).

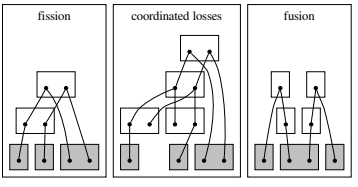


Fig. 3. Ambivalent explanations of fission events for two gene trees and three proteins

2 A Model for the Evolution of MDPs

To reconstruct MDP evolution, we require the composition of extant proteins and the phylogenetic relationships between domains of the same family. We now give an overview of the types of operations on MDPs and derive our model.

2.1 Evolutionary Events

We consider five macroevolutionary events (Fig. 2). Duplications and speciations are undistiguished *copy* events, as we do not rely on a species tree. *Fusion* describes the union of two ancestral MDP compositions via loss of terminal and initial segments or translocation. *Losses* originate from truncations due to premature stop codons or silencing of exons. Many perceived losses might be missing annotations as domain prediction has a high false-negative rate [16]. A *gain* is the introduction of the root of a domain tree and a *repeat* describes the addition of a domain (e. g. by tandem duplication). *Fission* of an MDP is a complex process requiring the gain of both a start and a stop site in the right order. The process has been hypothesized to involve reading frame shifts [17]. An alternative scenario for fission involves gene duplication with subsequent coordinated domain losses [18]. A third variant explains the observations by a fusion process (see Fig. 3). We model fission by a combination of other basic events and the score of the optimal plexus topology happens to be invariant to the explicit series of events.

2.2 The Plexus

Generally, a plexus is a meshwork of branching and rejoining strands, e. g. a network of blood vessels or neurons as in choroid plexus or solar plexus. It connotates that the strands have a direction as in blood flow or action potential propagation. We use the term to describe the aggregation and segregation of phylogenetic domain trees such that the nodes correspond to extant and reconstructed ancestral MDPs. We assume domain trees to be rooted and fully binary for this work.

An expanded plexus is constructed by linking the basic evolutionary events (see Fig. 1, left side and Fig. 2). This results in a *directed acyclic graph* (DAG) whose nodes are sets of domain tree nodes and whose edges are made up by the edges of the domain trees; to avoid confusion, we call the plexus' nodes *blocks* and its edges *arcs*. As shown in Fig. 1, the trees in the plexus are *not* necessarily the input trees but display them. Any subtree that contains no node in the plexus' leaves will not be in a subgraph of the input trees. Also, any root that has only one child will not be found in the reconstructed domain trees. The dashed lines in Fig. 1 (left side) represent such non-reconstructable edges. There is an infinite number of plexūs with identical compact counterparts, thus displaying the same input domain trees. The only plexus we can actually reconstruct is the one we obtain by deleting the non-reconstructable subtrees from the expanded plexus by removing any nodes in non-leaf blocks that have a combined in- and out-degree ≤ 1. On this reconstructable plexus, we can apply a scoring scheme that approximates the number of MDP evolution events.

The reconstructable plexus is still not very handy as it has infinitely many possible topologies. We can however restrict the topologies to a finite number by requiring that each block must contain at least one node with out-degree 2, which makes the number of tree nodes an upper bound to the number of blocks. By contracting out-arcs of blocks containing only nodes of out-degree 1, we transform a reconstructable into a compact plexus. In Fig. 1 (middle), these are the arcs made up by the dashed edges.

With the definition of the compact plexus, the problem is reduced to partitioning domain tree nodes. It is infeasible to evaluate all potential partitions and we have developed a heuristic to find the best scoring topology. In the following section, we will give a more rigid formalization in order to derive the scoring scheme and the heuristic.

3 Reconstruction of the Compact Plexus

3.1 Basic Definitions and Notation

Let $G := (V, E)$ be a DAG. We denote the in-degree and the out-degree of a node $v \in G$ by $\deg^-(v)$ and $\deg^+(v)$ respectively. The *edge contraction* of an edge $(v, w) \in E$ is achieved by first identifying v with w, and then deleting the resulting loop. For nodes $u, w \in V$ and $j \in \mathbb{Z}^+ \cup \{\infty\}$ we (i) write $u \sim_j w$, if $u \neq w$ and there is a path from u to w of at most j edges in G, and (ii) define

$u \sim_{-j} w := w \sim_j u$. If $u \sim_k w$ and $k > 1$, then we call u a *predecessor* of w, and w a *successor* of u. In the case $k = 1$, we use the terms *direct predecessor* and *direct successor* accordingly. We say that u and w are *connected* if $u \sim_\infty w$. Let $k, l \in \mathbb{Z} \cup \{-\infty, \infty\}$, then we define the *k-neighborhood* of a set $U \subseteq V$ to be $\mathrm{N}^k(U) := \{v \in V \mid \exists u \in U : v \sim_k u\}$, and $\mathrm{N}^{l,k}(U) := \mathrm{N}^k(\mathrm{N}^l(U))$. For instance, given a directed path $a \to b \to c$, $N^{1,1}\{a\} = N^1\{b\} = \{c\}$.

3.2 Plexus and Evolutionary Events

Let $G := (V, E)$ be a DAG. We call the graph $P(G) := (\mathcal{V}, \mathcal{E})$ a *plexus over* G if the following conditions are satisfied: (i) $P(G)$ is a DAG, (ii) $\mathcal{V}$ is a partition of V such that no pair of nodes in any $\mathcal{v} \in \mathcal{V}$ is connected in G, and (iii) $(\mathcal{a}, \mathcal{b}) \in \mathcal{E}$ iff there is a pair of nodes $a \in \mathcal{a}$ and $b \in \mathcal{b}$ for which $(a, b) \in E$.

We refer to plexus vertices as *blocks* and edges between blocks as *arcs*. Plexus notation is identical to graph notation, but is distinguished for clarity by *calligraphic script*. Blocks represent the composition of an MDP and arcs describe their inheritance relations. $\mathrm{C}_{\mathcal{w}}(\mathcal{v}) := \{p | p \in \mathcal{v} : \exists c \in \mathcal{w} : (p, c) \in E(G)\}$ is the set of nodes in block $\mathcal{v}$ that have children in block $\mathcal{w}$, and $\mathrm{P}_{\mathcal{v}}(\mathcal{w}) := \{c | c \in \mathcal{w} : \exists p \in \mathcal{v} : (p, c) \in E(G)\}$ the set of nodes in $\mathcal{w}$ with parents in $\mathcal{v}$.

Let $P := (\mathcal{V}, \mathcal{E})$ be a plexus. We call P *expanded* if for each of its blocks $\mathcal{b} \in \mathcal{V}$ either $(\mathcal{b}, \mathrm{N}^1\{\mathcal{b}\})$ or $(\mathrm{N}^{-1}\{\mathcal{b}\}, \mathcal{b})$ is an MDP evolution event. For brevity, a formal definition of MDP evolution events is omitted here but can be found in [19]. P is called *reconstructable* if no non-terminal block contains any node for which the sum of its in- and out-degree is less then 2 . A reconstructable plexus R is called the *reconstructable counterpart* of an expanded plexus P iff it can be obtained by subsequently deleting any non-terminal nodes that have an in- or out-degree of 0 and their incident edges. Let $\mathcal{e} := (\mathcal{v}, \mathcal{w}) \in \mathcal{E}$ be an arc such that $\forall v \in \mathrm{C}_{\mathcal{w}}(\mathcal{v}) : \deg^+(v) = 1$ and $\forall w \in \mathrm{P}_{\mathcal{v}}(\mathcal{w}) : \deg^-(w) = 1$. Let $\mathcal{e}$ be an arc such that $\forall e := (v, w) \in \mathcal{e} : \deg^+(v) = 1, \deg^-(w) = 1$. The operation of contracting all $e \in \mathcal{e}$ and merging $\mathcal{v}$ with $\mathcal{w}$ is called *arc contraction*. A plexus C is said to be *contracted* if it contains no contractible arcs, and *contracted counterpart* of a plexus P if it is contracted and can be obtained by contracting arcs in P. This is similar to the concept of *minors* in undirected graphs. A contracted plexus C is called the *compact counterpart* of a plexus P, if there is a reconstructable counterpart R of P such that C is a contracted counterpart of R.

3.3 Scoring Evolutionary Scenarios

To measure the quality of our reconstruction, we introduce a score on the compact plexus that considers evolutionary events by a unified criterion. Only losses, gains and fusions are events in which blocks connected by an arc contain nodes that are not related to any node in the other block. In contrast to copy and repeat, the direct successor blocks are intrinsically different from their predecessors. The number of these domains is therefore a good measure to model evolutionary changes.

Unfortunately, compactification imposes contraction to arcs in fusion, gain and loss, and hence to exactly those events that we consider to be of evolutionary importance. We can reconstruct them from the compact counterpart.

The number of losses accounting for a block v from all its ancestors is $\sum_{p\in \mathrm{N}^{-1}\{v\}} \left(|p| - |\mathrm{P}_p(v)|\right)$ which equals $\sum_{p\in \mathrm{N}^{-1}\{v\}} |p| - \sum_{p\in \mathrm{N}^{-1}\{v\}} |\mathrm{P}_p(v)|$.

The number of gains is equal to the number of domain trees and constant for all topologies, and therefore omitted. The remaining problem is to address the contraction of fusion arcs, which can increase the in-degree of a block. We can relate the number of fusion arcs in a plexus to its compact counterpart. Let P_E be an expanded plexus and P_C its compact counterpart. Then the number of fusion arcs in P_E equals $\sum_{b\in \mathcal{V}(P_C)} \max\{0, 2\cdot \deg^-(b) - 2\}$.

The order of fusions is lost during compactification but the number of domain changes depends on that order. Consider a block with in-degree 3 and predecessor blocks of size 1, 2 and 3. Combining 1 and 2 first creates an out-arc of size 3, and then merging with the third block creates an out-arc of size 6. In contrast, combining 2 and 3 first produces out-arcs of size 5 and 6, so the score would have to be 2 edges higher. In other words, there are reconstructable plexūs with different fusion sequences that have the same compact counterpart. As the real sequence of fusions is unknown, we use the mean number of nodes at the end of in-arcs, which defines the following *fusion cost* $\frac{\max\{2\cdot\deg^-(v)-2,0\}}{\max\{\deg^-(v),1\}} \cdot \sum_{p\in \mathrm{N}^{-1}\{v\}} |\mathrm{P}_p(v)|$.

This formula also holds in cases in which an arc involves a tandem repeat. Combining the above equations for losses, gains and fusions and summing up over all blocks yields the *plexus score* $S(P)$ as the score of its compact counterpart P_C as

$$S(P) = \sum_{v\in \mathcal{V}(P_C)} \sum_{p\in \mathrm{N}^{-1}\{v\}} \left(|p| + \left(1 - \frac{2}{\deg^-(v)}\right) \cdot |\mathrm{P}_p(v)| \right).$$

Note that $\max\{\ldots\}$ in the fusion cost formula only serves to avoid negative costs for root blocks. As the index set $\mathrm{N}^{-1}\{v\}$ is empty in this problematic case, the formula is simplified.

Given the scoring scheme above, we now define the following problem:

Problem 1 (plexus reconstruction)
Instance: A set T of fully binary domain trees and a partition $\mathcal{L}$ of their combined leaf set such that each set block corresponds to a known MDP composition.
Find: a compact plexus P in which $\mathcal{L}$ is the leaf block set and which displays T such that the plexus score $S(P)$ is minimal.

4 Heuristic

The definition of our reconstruction problem above applies to input trees free of errors. It is unknown whether there is an analytical solution within acceptable run-time complexity for undistorted input. A thorough evaluation would be

worthwhile but is beyond the scope of this work. In real applications the input trees typically contain numerous wrong splits. Trees built on domains use less information than trees on full-length proteins simply because they are shorter.

Our method works in three steps, which correspond to the events we want to minimize. In the initial *block merging* step, we merge non-leaf blocks according to a compatibility criterion that asserts an out-degree ≤ 2 (d-compatibility), and allows only for compositions that resemble those of the input (t-reconcilability). The latter also tolerates compositions that are close to the observed with additional domains to account for false tree splits but mainly reduces the number of *fusions*. In the second step (*tree correction*), we attempt to correct the placement of tree nodes based on the preliminary topology of the plexus to minimize the number of *coordinated losses*, i. e. two domains of the same family that each have only one child, but in different direct successor blocks, as shown for the solid domain in Fig. 4. In the final *path detachment* step, we separate sub-blocks consisting of nodes that are placed too high in the plexus by previous steps, which reduces *unnecessary losses*.

4.1 Block Merging

To reduce the number of fusion events we merge non-leaf blocks. To restrict the merged blocks' out-degree to ≤ 2, we use transitive reduction of arcs. An arc (v, w) is a *transitive arc* (v, w) if $w \in \mathrm{N}^k\{v\}$ for any $k > 1$. The path of a transitive arc (v, w) is given by $(v, \mathfrak{b}_1, \ldots, \mathfrak{b}_k, w)$. One can insert k nodes in each edge $e = (v_v, v_w)$ in (v, w), thus creating paths $(v_v, v_1, \ldots, v_k, v_w)$. Placing each v_i into $\mathfrak{b}_i$ reduces the out-degree of v by 1. Consequently, blocks are *reducible* if their out-degree can be reduced by transitive reduction of outgoing arcs. This holds iff $\mathrm{N}^1\{\mathfrak{b}\} \cap \mathrm{N}^{1,\infty}\{\mathfrak{b}\} \neq \emptyset$.

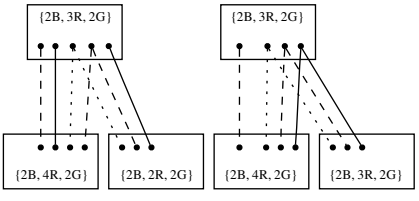


Fig. 4. Composition profiles. In the reconstruction of an ancestral block we show a typical artifact from errors in the phylogenetic reconstruction that leads to additional domains in the ancestral block, here the nodes of the tree with solid edges in the left figure. Despite the left variant containing two copies of the solid domain family the outdegree-profile for both variants is identical.

For any pair of blocks it is necessary to know whether there is a transitive reduction to the merged block such that its out-degree does not exceed 2. Assume that the blocks are irreducible, as we could always apply transitive reduction before a merge.

Theorem 1 (minimal out-degree). *Let v and w be two irreducible blocks such that $v \notin \mathrm{N}^{\infty}\{w\} \cup \mathrm{N}^{-\infty}\{w\}$. Let χ be a block obtained by merging v and w. Then the minimal out-degree $\deg^{\triangleleft}(\chi)$ that can be obtained by a sequence of transitive reductions to χ is $\deg^{\triangleleft}(\chi) := \left|\mathrm{N}^1\{v\}\right| + \left|\mathrm{N}^1\{w\}\right| + \left|\mathrm{N}^1\{v\} \cap \mathrm{N}^1\{w\}\right| - \left|\mathrm{N}^1\{v\} \cap \mathrm{N}^{\infty}\{w\}\right| - \left|\mathrm{N}^{\infty}\{v\} \cap \mathrm{N}^1\{w\}\right|$.*

A proof is omitted for brevity and can be found in [19]. We can find all pairs of blocks that can be merged without violating neither out-degree d nor compact plexus properties by the following criterion:

Definition 1 (d-compatibility). *Two irreducible blocks v and w are called* d-compatible *if $\deg^{\triangleleft}(v \cup w) \leq d$, i. e. one can obtain a block with an out-degree of at most d by merging v and w and applying a sequence of transitive reductions to the merged block. However, if either of them is a leaf, or $v \notin \mathrm{N}^{\infty}\{w\} \cup \mathrm{N}^{-\infty}\{w\}$ (blocks are related), then they are incompatible.*

Definition 2 (reduction cost r). *Let v be a block with an out-degree greater than 0. The* reduction cost *of an outgoing arc pointing to block $w \in \mathrm{N}^1\{v\}$ is the smallest $k > 0$ for which $w \in \mathrm{N}^k(\mathrm{N}^1\{v\} \setminus \{w\})$, or 0 if there is no such k, i. e. the arc is not transitive and thus cannot be removed. The reduction cost $r(v)$ of a block is the sum of costs of all its outgoing arcs.*

An *a-priori* set of candidates excluding all blocks that cannot be compatible with the current block ensures a tractable solution space. Only the direct predecessors of all successors of each block v need to be checked for compatibility.

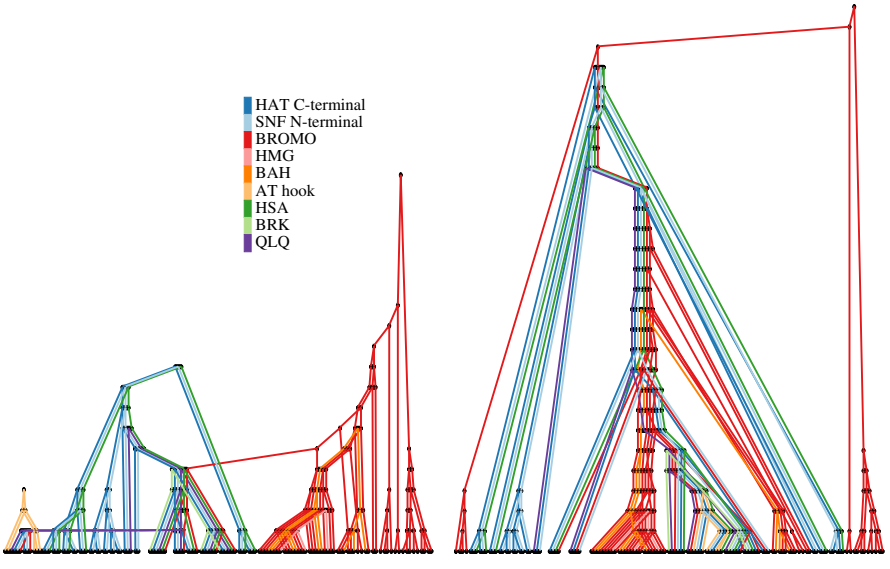
2-compatibility alone leads to domain compositions that do not resemble recent MDPs, leading to many losses as seen in Fig. 5(b). Many compatibilities arise merely by chance or by false tree splits. We therefore ensure that blocks *resemble recent compositions* by the following:

Definition 3 (composition profile). *Let $M = \{d_1, \ldots, d_k\}$ be a set of nodes in a block. M is partitioned into subsets $\{F_1, \ldots, F_m\}$ of nodes that belong to the same input tree, the set of families is denoted by representants $p := \{[\![F_1]\!], \ldots, [\![F_m]\!]\}$. Let $m(\cdot) : [\![F_i]\!] \to \mathbb{N}$ be the mapping $m([\![F_i]\!]) = 2 \cdot |\{n | n \in F_i, \deg^{+}(n) = 0\}| + \sum_{d \in F_i} \deg^{+}(d)$. Then (p, m) is called the* composition profile *of M.*

Definition 4 (t-reconcilability). *A profile p_1 is called* t-reconcilable *to a profile p_2 if $\forall [\![F_i]\!]_1 \in p_1 : \exists [\![F_i]\!]_2 \in p_2 : [\![F_i]\!]_1 = [\![F_i]\!]_2, m([\![F_i]\!]_1) \leq m([\![F_i]\!]_2) + t$, where t is a non-negative integer describing a chosen tolerance value.*

Simply put, a value is assigned to each domain family that describes how often a domain of this family occurs in a composition. Those without children are given the same value as those with two children, those with just one child are weighted half. A block in a compact plexus will either contain only nodes without children, or no node without children. The reasoning behind this definition is illustrated in Fig. 4: on the right the upper block resembles its left direct successor, whereas on the left it contains one solid domain more than any of its direct successors.

Both predecessor blocks have the same profile though, since both solid domains have only one out-edge each. We might call this a *coordinated loss* of the solid domain; this will either be caused by a tree root being placed in the block above, but will often occur due to false tree splits. These might introduce disruptions to the optimal topology. t-reconcilability aims to compensate this, while providing a concept of similarity to recent compositions. One should choose t to be small to avoid meaningless ancestral compositions and large loss counts, but $t = 0$ assumes that the topologies of all input trees are correct, which will rarely be the case. $t = 1$ yielded the best results in our hands. Combining d-compatibility and t-reconcilability provides us with a criterion for the merges to prefer and to avoid:



(a) $S(P) = 45$ (b) $S(P) = 61$

Fig. 5. Two compact plexūs of domains in histone acetyltransferase complexes. In (a) we used 1-reconcilability, tree correction and path detachment, predicting a late fusion event of the BROMO domain. In (b) only d-compatibility was used, resulting in a single-source plexus with multiple losses and compositions that do not resemble extant MDPs. Note that, among others, the BROMO domain fusion is placed much too high (horizontal edge at top). Labeled high-resolution figures can be downloaded for detailed analysis from `http://genome.cs.iastate.edu/CBL/ISBRA10/thesis.zip`

Definition 5 (d-t-distance). *If two blocks v, w are d-compatible and the profile of $v \cup w$ is t-reconcilable to a profile of any input composition, their* d-t-distance $c(v, w)$ *is $r(v \cup w)$, otherwise it is ∞.*

Initially, we alternate between transitive reduction of all blocks and merging the two blocks with the shortest d-t-distance, until there is no pair whose distance is $< \infty$. We avoid merging repeat blocks with copy blocks and thus violating

compact plexus properties by inserting an additional block in the copy block's out-arc and merging it with the repeat block.

4.2 Tree Correction

The above procedure can introduce new blocks below old ones, thus pushing some tree nodes higher during block merges, thus *stretching* subtrees and give rise to additional losses. To compensate for this, we introduce *tree correction*: two tree nodes can be merged if they are in the same block, have the same parent node, and the out-degree of the merged node is ≤ 2 or can be reduced by recursively merging child nodes respectively. Root nodes must not be merged. If the common parent of two nodes being merged is a root node, it can be deleted after the merge, as the merged node will be a new root.

4.3 Path Detachment

Let there be any arc path $((a, b), (b, c))$. If all nodes in b that have parents in a (i. e. $\mathrm{P}_a(b)$) only have children in c (i. e. $\mathrm{N}^1(\mathrm{P}_a(b)) \subseteq c$), then this induces unnecessary domain losses, as the composition b is only supported by one direct successor. One can therefore split the block b into $\mathrm{P}_a(b)$ and $b \setminus \mathrm{P}_a(b)$, and apply this procedure recursively to their direct successors, thus reducing the number of loss events. After that, applying arc contraction ensures a compact plexus.

4.4 Time Complexity

The merge step dominates the running time. To decide which blocks to merge, one has to calculate path distances between their direct successors. A plexus is a DAG with all arcs having the same weight. Shortest paths are thus subgraphs of a breadth-first search tree. One has to create such a tree $|\mathcal{R}|$ times with $\mathcal{R}$ being the set of root blocks, so the time complexity of finding all pairs shortest paths is in $O\{|\mathcal{R}| \cdot (|\mathcal{V}| + |\mathcal{E}|)\}$. Since any block has two out-arcs at most, this is in $O\{|\mathcal{R}| \cdot |\mathcal{V}|\}$. Finding the smallest d-compatibility by pairwise comparison takes time in $O\{|\mathcal{V}|^2\}$. With $\mathcal{L}$ being the leaf set, $|\mathcal{L}|$ is the number of profiles one has to check, so the time for finding the d-t-closest pair lies in $O\{|\mathcal{R}| \cdot |\mathcal{V}| + |\mathcal{L}| \cdot |\mathcal{V}|^2\}$. As the number of blocks decreases with every merge, one has to perform this $\leq |\mathcal{V}|$ times at most, if all distances are recalculated in each step. The time complexity of the merge step is $O\left\{\sum_{v=1}^{|\mathcal{V}|} \left(|\mathcal{R}| \cdot v + |\mathcal{L}| \cdot v^2\right)\right\} \subseteq O\left\{\sum_{v=1}^{|\mathcal{V}|} v^3\right\} \subseteq O\{|\mathcal{V}|^4\}$ Both tree correction and path detachment traverses subtrees of the plexus but this is linear and depends on the number of blocks.

5 Application

We obtained identical results for the examples given in [13] (data not shown). To test our heuristic on proteins assembling to histone acetelyase complexes in *H. sapiens, D. melanogaster, S. cerevisiae, S. pombe*, and *A. thaliana*, we selected the proteins containing the BROMO, the N-terminal SNF2 and the

C-terminal conserved helicase domains of the histone acetyltransferases as identified in PFAM [20]. Domains were aligned with `hmmalign` of the HMMer package. Maximum Likelihood trees were constructed using `PhyML` [21]. Notung 2.6 was used to root the domain trees [22]. The input plexus had a score of 166, the result obtained heuristically scored 45 (Fig. 5(a)).

6 Conclusion and Outlook

We have presented an approach to reconstruct ancestral multi-domain proteins using plexūs. A suitable scoring scheme together with a heuristic allows finding near-optimal solutions.

Improvements to d-compatibility could enhance the use of real data and extending it to weighted paths would allow the use of bootstrap-valued DAGs instead of trees to deal with ambiguity in the phylogenetic signal. It could also be modified to handle non-binary or unrooted trees. A compatibility constraint based on domain order would be helpful in separating true losses from missing annotations.

As seen in the heuristic, random compatibility is an important issue. We adress it by t-reconcilability, path-detachment and tree correction, but the development of a statistical model that assigns a p-value to a plexus topology would be worthwhile. Constraint optimization approaches might allow for considerable speedup in the implementation and possibly even find an optimal solution.

Acknowledgements

We thank M. Homilius, I. Kel, C. Standfuß and S. Thieme for sharing data. This work was supported in part by the NSF AToL program DEB 0830012.

References

1. Doolittle, R.F.: The multiplicity of domains in proteins. Annual Review of Biochemistry 54, 287–314 (1995)
2. Koonin, E.V., Altschul, S.F., Bork, P.: BRCA1 protein products... Functional motifs... Nature genetics 13(3), 266 (1996)
3. Ekman, D., Björklund, Å.K., Frey-Skött, J., Elofsson, A.: Multi-domain Proteins in the Three Kingdoms of Life: Orphan Domains and Other Unassigned Regions. Journal of Molecular Biology 348(1), 231–243 (2005)
4. Basu, M.K., Carmel, L., Rogozin, I.B., Koonin, E.V.: Evolution of protein domain promiscuity in eukaryotes. Genome Res. (2008) gr.6943508+
5. Song, N., Joseph, J.M., Davis, G.B., Durand, D.: Sequence similarity network reveals common ancestry of multidomain proteins. PLoS computational biology 4(5) (2008)
6. Apic, G., Gough, J., Teichmann, S.A.: An insight into domain combinations. Bioinformatics 17(suppl. 1) (2001)
7. Yang, S., Doolittle, R.F., Bourne, P.E.: Phylogeny determined by protein domain content. Proceedings of the National Academy of Sciences 102(2), 373–378 (2005)

8. Björklund, Å.K., Ekman, D., Light, S., Frey-Skött, J., Elofsson, A.: Domain Rearrangements in Protein Evolution. Journal of Molecular Biology 353(4), 911–923 (2005)
9. Przytycka, T., Davis, G., Song, N., Durand, D.: Graph Theoretical Insights into Evolution of Multidomain Proteins. Journal of Computational Biology 13(2), 351–363 (2006)
10. Fong, J.H., Geer, L.Y., Panchenko, A.R., Bryant, S.H.: Modeling the Evolution of Protein Domain Architectures Using Maximum Parsimony. Journal of Molecular Biology 366(1), 307–315 (2007)
11. Ciccarelli, F.D., von Mering, C., Suyama, M., Harrington, E.D., Izaurralde, E., Bork, P.: Complex genomic rearrangements lead to novel primate gene function. Genome research 15(3), 343–351 (2005)
12. Lucas, J.I., Arnau, V., Marín, I.: Comparative genomics and protein domain graph analyses link ubiquitination and RNA metabolism. J. Mol. Biol. 357(1), 9–17 (2006)
13. Behzadi, B., Vingron, M.: Reconstructing Domain Compositions of Ancestral Multi-domain Proteins, pp. 1–10. Springer, Heidelberg (2006)
14. Huson, D.H., Bryant, D.: Application of phylogenetic networks in evolutionary studies. Molecular Biology and Evolution 23(2), 254–267 (2006)
15. Moret, B.M.E., Nakhleh, L., Warnow, T., Linder, C.R., Tholse, A., Padolina, A., Sun, J., Timme, R.: Phylogenetic Networks: Modeling, Reconstructibility, and Accuracy. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1(1), 1–12 (2004)
16. Moore, A.D., Björklund, Å.K., Ekman, D., Bornberg-Bauer, E., Elofsson, A.: Arrangements in the modular evolution of proteins. Trends in Biochemical Sciences 33(9), 444–451 (2008)
17. Snel, B., Bork, P., Huynen, M.: Genome evolution: gene fusion versus gene fission. Trends in Genetics 16(1), 9–11 (2006)
18. Wang, W., Yu, H., Long, M.: Duplication-degeneration as a mechanism of gene fission and the origin of new genes in Drosophila species. Nature Genetics 36(5), 523–527 (2004)
19. Wiedenhoeft, J.: Phylogenetic Reconstruction of Ancestral Multidomain Proteins (2009), BSc thesis, http://genome.cs.iastate.edu/CBL/ISBRA10/thesis.zip
20. Finn, R.D., Tate, J., Mistry, J., Coggill, P.C., Sammut, S.J., Hotz, H.R., Ceric, G., Forslund, K., Eddy, S.R., Sonnhammer, E.L.L., Bateman, A.: The Pfam protein families database. Nucleic acids research 36(Database issue), D281–D288 (2008)
21. Guindon, S., Gascuel, O.: A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. Systematic biology 52(5), 696–704 (2003)
22. Durand, D., Halldorsson, B.V., Vernot, B.: A hybrid micro-macroevolutionary approach to gene tree reconstruction. Journal of Computational Biology 13(2), 320–335 (2006)

Local Structural Alignment of RNA with Affine Gap Model

Thomas K.F. Wong, Brenda W.Y. Cheung, T.W. Lam, and S.M. Yiu

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

Abstract. Predicting new non-coding RNAs (ncRNAs) of a family can be done by aligning the potential candidate with a member of the family with known sequence and secondary structure. Existing tools either only consider the sequence similarity or cannot handle local alignment with gaps. In this paper, we consider the problem of finding the optimal local structural alignment between a query RNA sequence (with known secondary structure) and a target sequence (with unknown secondary structure) with the affine gap penalty model. We provide the algorithm to solve the problem. Based on a preliminary experiment, we show that there are ncRNA families in which considering local structural alignment with gap penalty model can identify real hits more effectively than using global alignment or local alignment without gap penalty model.

Keywords: Local structural alignment, Affine gap, non-coding RNA.

1 Introduction

A non-coding RNA (ncRNA) is a RNA molecule that does not translate into proteins. It has been shown to be involved in many biological processes [1,2,3,4]. The number of ncRNAs within the human genome was underestimated before, but recently some databases reveal over 212,000 ncRNAs [5] and more than 1,300 ncRNA families [6]. Large discoveries of ncRNAs and their families show the possibilities that ncRNAs may be as diverse as protein molecules [7]. Identifying ncRNAs is an important problem in biological study. However, it is time consuming and there is no effective method to identify ncRNAs in a laboratory, predicting ncRNAs based on known ncRNAs using comparative computational approach is one of the promising directions to identify potential candidates for further verification.

Most of the computational approaches are based on the observation that if two different ncRNA molecules are in the same family (with similar biological functions), they usually exhibit similar sequences as well as secondary structures. One common approach [8,9,10] is as follows. We pick an ncRNA member of a family with known sequence and secondary structure (referred as the query), scan along a genomic sequence and for each possible region (referred as the target), perform an alignment between the query and the target to obtain a similarity measure to decide if the region is a potential ncRNA candidate for that family.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 191–202, 2010.

The similarity measure may only base on the sequence or both the sequence and secondary structure (the latter case is referred as *structural alignment*). Along this direction, there are some approaches [11,12,13,14] that make use of secondary structure prediction tools to predict the secondary structure to be formed by the target assuming that it is an ncRNA before performing the alignment. The accuracy may, however, depend on the accuracy of the secondary structure prediction tools.

Instead of using one member of a family, some other approaches [15] use a set of ncRNAs from the same family to train a model (e.g. covariance model). Then, using this model to scan a genomic sequence to identify potential regions that are ncRNA candidates of that family. What information (sequence similarity and/or secondary structure) to be captured from the known ncRNAs depends on how we define the model. However, in some cases, we may not have enough known members in a family to train a model. In this paper, we focus on the problem that uses one known member as the query and align it with a target sequence. We remark that there are also other computational methods that identify ncRNAs without using known members in a family. For example, some try to identify ncRNAs by considering the stability of secondary structures formed by the substrings of a given genome [16]. This method may not be very effective because a random sequence with high GC composition also allows an energetically favorable secondary structure [17]. So, the comparative approach we described in the above is still one of the most popular approaches.

The core idea behind all comparative approaches is to compute the similarity between the query (known member(s)) and the target (each possible region in the genomic sequence to be investigated). Some only consider sequence similarity which may not work well for families in which members do not have high sequence similarity (e.g. members of RF00017 in Rfam 9.1 [6] only have 39% sequence similarity). For example, Gotohscan [8] considers semi-global alignment with affine gap penalty according to the sequence similarity only. For those also consider the similarity of secondary structure, they usually require the whole sequence of the query to be aligned with the whole sequence of the target (referred as *global* alignment in the community) [10]. However, similar to the protein sequence, the ncRNAs in the same family may not have similar sequence or structure for the whole sequence but only for the substrings of them (those supposed to be the functional parts), especially when they belong to species with long evolutionary distance apart. Fig. 1 shows one of these examples. It shows the multiple sequence alignment between some members of the family RF01051 in Rfam 9.1 database. The two circled members (i.e. AAUO01000012 and AAXYO1000014) are not quite similar if we consider the global alignment. Also, for the subregions that they look similar (i.e. the circled region), there exist large insertion/deletion (gaps). There are also evidences that gaps may be common in ncRNA homologs [18]. Considering local structural alignment with gap model seems to be more appropriate for predicting new members for some ncRNA families. [9] consider some restricted cases of local alignment according to the query structure. Another work that also consider local alignment is [11], but they cannot handle gaps.

Fig. 1. Multiple sequence alignment of some seed members of the family RF01051 from Rfam 9.1 database. The red and blue highlighted are the base-pair regions. All sequences are aligned according to their structures. If the two circled sequences are selected as query and target, the circled region is the conserved local region between them, in which there exists long gap inside.

We consider the following problem. Given a query sequence together with its secondary structure, we try to identify the substring in the given target sequence (with unknown secondary structure) that can align to a substring in the query sequence with the highest structural similarity score based on the affine gap model (see Section 2 for formal definitions). We assume that the secondary structures of the ncRNAs are regular, that is, they do not have pseudoknots (no two base pairs crossing each other). This type of ncRNAs is found to be the most abundant in existing databases. We consider all possible substrings of the query sequence, even for those substrings that cover only one of the end points of some base pairs in the structure.

Our result: We propose a local structural alignment algorithm with affine gap model which assumes the secondary structure of the query is known while that of the target sequence is unknown. The time complexity of our algorithm is $O(mn^3)$ which is the same as the best algorithm for global alignment for this problem where m, n are the lengths of the query and the target, respectively. We evaluated our algorithm using real data from Rfam database. According to the preliminary experiment, it shows that using local structural alignment algorithm with affine gap model is more effective to distinguish real members from false hits for those families in which members have variable sizes of hairpins, loops or stems when compared to global alignment or local alignment without gap model.

2 Preliminaries

An ncRNA molecule can be regarded as a sequence of four characters $\{A, C, G, U\}$, each character is referred as a base. Some of these bases may form pairs (linked up by a hydrogen bond) with some restrictions such as each base can only pair up with at most one other base and only complementary bases can form a pair (e.g. (A, U), (C, G), (U,G)). The set of base pairs formed by the molecule is referred as its *secondary structure*.

Formally speaking, let $S = s_1 s_2 \dots s_m$ be a length-m ncRNA sequence where $s_i \in \{A, C, G, U\}$ for $1 \leq i \leq m$ and M be the secondary structure of S. M is represented as a set of base pair positions. i.e. $M = \{(i,j) | 1 \leq i < j \leq m, (s_i, s_j)$ is a base pair$\}$. If (s_i, s_j) is base pair, then $(s_i, s_j) \in \{(A,U), (C,G), (G,C), (G,U), (U,A), (U,G)\}$. Let $M_{x,y} \subseteq M$ be the set of base pairs within the subsequence $s_x s_{x+1} ... s_y$, $1 \leq x < y \leq m$, i.e., $M_{x,y} = \{(i,j) \in M | x \leq i < j \leq y\}$. Note that if $(i,j) \in M$ and only i or j inside the region $[x...y]$, then $(i,j) \notin M_{x,y}$. We assume that there is no two base pairs sharing the same position, i.e., for any $(i_1, j_1), (i_2, j_2) \in M$, $i_1 \neq j_2$, $i_2 \neq j_1$, and $i_1 = i_2$ if and only if $j_1 = j_2$.

A regular structure is the structure in which there does not exist any two base pairs crossing each other. The formal definition is as follows:

Definition 1. *$M_{x,y}$ is a* regular structure *if there does not exist two base pairs $(i,j), (k,l) \in M_{x,y}$ such that $i < k < j < l$ or $k < i < l < j$.*

Note that an empty set is also considered as a regular structure.

3 Problem Definition

3.1 Structural Alignment with Affine Gap Model

Let $S[1...m]$ be a query sequence with known secondary structure M, and $T[1...n]$ be a target sequence with unknown secondary structure. S and T are both sequences of {A,C,G,U}. A structural alignment between S and T is a pair of sequences $S'[1...r]$ and $T'[1...r]$ where $r \geq m, n$, S' is obtained from S and T' is obtained from T with spaces inserted to make both of the same length. A space cannot appear in the same position of S' and T'. A maximal consecutive set of ℓ spaces in either S' or T' is referred as a gap of length ℓ. The score of the alignment (with affine gap penalty model), which determines the sequence and structural similarity between S' and T', is defined as *score* =

$$\sum_{\substack{1 \leq i \leq r \text{ s.t.} \\ S'[i], T'[i] \neq \text{`\_'}}} \gamma(S'[i], T'[i]) + \sum_{\substack{i,j \text{ s.t. } \eta(i), \eta(j) \in M, \\ S'[i], S'[j], T'[i], T'[j] \neq \text{`\_'}}} \delta(S'[i], S'[j], T'[i], T'[j]) - (h(k) + s(l))$$

where $\eta(i)$ is the corresponding position in S according to the position i in S'; $\gamma(u_1, u_2)$ and $\delta(u_1, u_2, v_1, v_2)$ where $u_1, u_2, v_1, v_2 \in \{A, C, G, U\}$, are scores for character similarity and for base pair similarity respectively; k and l is the number of gaps and the total length of all gaps; h and s is the gap starting and extending penalty.

Definition 2. An optimal global structural alignment *between S and T is to a structural alignment of S and T such that the alignment score is maximum.*

Let $S[x...y]$ where $1 \leq x, y \leq m$ be a substring of S with secondary structure $M_{x,y}$ (where $S[x...y]$ is an empty string with empty structure if $x > y$). Similarly, let $T[x'...y']$ where $1 \leq x', y' \leq n$ be a substring of T (where $T[x'...y']$ is an empty string if $x' > y'$).

Definition 3. An optimal local structural alignment *between* S *and* T *is a global structural alignment between two substings of* S *and* T, $S[x...y]$ *and* $T[x'...y']$ *where* $1 \leq x, y \leq m$ *and* $1 \leq x', y' \leq n$ *of* S *and* T *such that the alignment score between them is maximum over all possible substrings.*

Given S (with known secondary structure) and T (with unknown structure), we want to compute an optimal local structural alignment with affine gap penalty between S and T.

4 Experimental Results

The details of the algorithm for solving the problem will be given in Section 5. In this section, we evaluate the resulting algorithm and show that considering local structural alignment with affine gap model can improve the effectiveness of locating ncRNAs for the families in which members may have variable size of hairpins, loops or stems when compared to using global alignment [10], local alignment without gap penalty model and Gotohscan [8]. Note that the differences in size of hairpins, loops or stems represent gaps in the corresponding sequences.

To explicitly test the algorithm, we selected four ncRNA families: RF00386, RF00643, RF00661 and RF01051, in which the members have variable sizes of hairpins, loops or stems. We construct our testing cases based on real ncRNAs as follows. For each family, we first select a seed member[1] as the query sequence Q. To demonstrate the effectiveness of the affine gap model, we select the longest seed member as this query sequence. We then created a long random sequence with even distribution of four characters $\{A, C, G, T\}$ to simulate a long genome. The length of this long random sequence is around ten times of the total length of all the seed members of the family. Finally, we embedded all the seed members of the family (except the one chosen as query) into this long random sequence in arbitrary positions. This resulting sequence is our T.

For every region in T with length similar to that of the query sequence [2], we compute the structural alignment score of the region and the query sequence. We use the same scoring scheme as in [9] and set the gap starting penalty (h) and gap extension penalty (s) to be 5 and 0.2, respectively. The details of the families including the sequence selected as the query, the length of the sequence, and the number of seed members in each family are given in Table 1.

We compare our algorithm with the global structural alignment [10], local structural alignment without affine gap model and Gotohscan [8]. Gotohscan was used to locate ncRNAs candidates on Trichoplax adhaerens by using single real ncRNA as query. It was designed to check only sequence similarity with affine gap model. Since the global structural alignment software is not available, we implemented both global and local without affine gap algorithms. For Gotohscan,

[1] In Rfam database, there is a set of reliable members which are regarded as seed members. In our experiments, we only use seed members.

[2] We set the length of each region equals the length of the query plus 20.

Table 1. The details of the ncRNA families used in the experiments

Family	Query Sequence ID	Length	Number of seed members
RF00386	AF363455.1/1-122	122	160
RF00643	AASG02000279.1/67999-67862	138	46
RF00661	AC154049.1/4734-4855	122	40
RF01051	AAUO01000012.1/70652-70532	121	169

we downloaded the version 1.3 from the website. We assume that regions other than the seed members of the family are false hits as they are likely not to be members of the family. To compute the effectiveness of our method, we set the threshold as the maximum score of the false hits. We assume that the method finds a real hit if the score of the region is larger than this threshold. Thus a real hit will be missed if the computed score is smaller than or equal to this threshold. We also try different thresholds and the results are similar. Table 2 summarizes the result when using different algorithms to locate the other ncRNA members along the genome. The % of misses when using Gotohscan is 22.0%-88.9%; global alignment is 10.1%-84.6%; local alignment without affine gap model is 5.7% - 69.2%; local alignment with affine gap model is 0.0% - 28.9%. The result shows that the local structural alignment algorithm with affine gap model is more effective than the other algorithms in these families.

Figure 2 shows the comparison on score distribution of real hits (i.e. real members) and false hits for the family RF00661 between different algorithms. It shows that the local structural alignment algorithm with affine gap penalty can increase the difference between the scores of real hits and the scores of false hits compared with the other methods, and so it has a higher distinguishing power to identify the real ncRNA members along the long genome sequence.

Table 2. Summary of comparison on results between global alignment, local alignment without gap penalty and local alignment with affine gap penalty

Family	Number of members	Number of misses							
		Gotoh-scan [8]	%	Global [10]	%	Local	%	Local with affine gap	%
RF00386	159	35	22.0%	16	10.1%	9	5.7%	0	0.0%
RF00643	45	40	88.9%	30	66.7%	13	28.9%	13	28.9%
RF00661	39	33	84.6%	33	84.6%	27	69.2%	10	25.6%
RF01051	168	121	72.0%	99	58.9%	83	49.4%	36	21.4%

Our program take around 15 seconds for performing local structural alignment with affine gap model between query and target of around 150 bases long, and around 30 seconds for 200 bases long. We tested the program on a machine with 2.4GHz dual-core CPU and 8G memory.

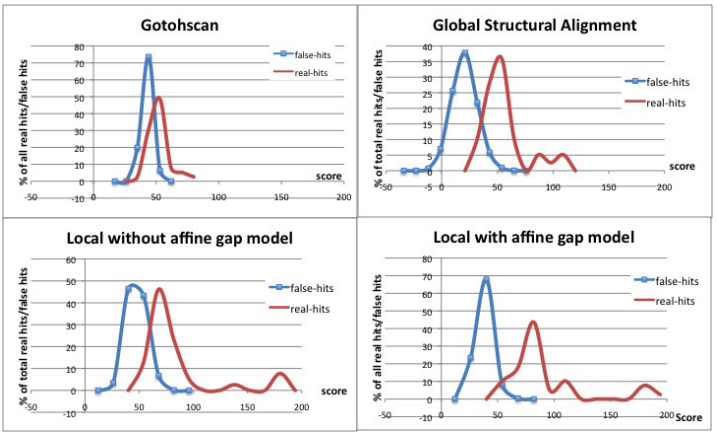


Fig. 2. Comparison on score distribution of real hits and false hits for the family RF00661 between (1) Gotohscan, (2) global structural alignment algorithm, (3) local structural alignment algorithm without affine gap model, and (4) local structural alignment algorithm with affine gap model

5 Method

We develop a dynamic programming algorithm to solve the problem. Before we describe the method, we would like to define some variations of alignments which will be used in our algorithm. Let $S[1...m]$ be the query sequence with known structure M and $T[1...n]$ be the target sequence with unknown structure.

Definition 4. Optimal prefix-global structural alignment *between* $S[1...m]$ *and* $T[1...n]$ *is to find a prefix* $S[1...y]$ *where* $0 \leq y \leq m$ *(i.e.* S *is an empty string when* $y = 0$*) such that the score of the optimal global structural alignment between the prefix* $S[1...y]$ *and* $T[1...n]$ *is maximum.*

Definition 5. Optimal suffix-global structural alignment *between* $S[1...m]$ *and* $T[1...n]$ *is to find* $S[x...m]$ *where* $1 \leq x \leq m+1$ *(i.e.* S *is an empty string when* $x = m+1$*) such that the score of the optimal global structural alignment between the suffix* $S[x...m]$ *and* $T[1...n]$ *is maximum.*

Definition 6. Optimal semi-global structural alignment *between* $S[1...m]$ *and* $T[1...n]$ *is to find a substring* $S[x...y]$ *where* $1 \leq x, y \leq m$ *such that the score of the optimal global structural alignment between the substring* $S[x...y]$ *and* $T[1...n]$ *is maximum.*

Let the affine gap model be $h + sL$, where h is the gap opening penalty, s represents a gap extension penalty, and L denotes the length of gap. Our method consists of two steps. In the first step, we compute the optimal semi-global structural alignment between S and all possible substrings of T. In the second step, we obtain the optimal local structural alignment between S and T resulted in the first step.

Define $A(p, q, e, f)$ be the score of the optimal *semi-global* structural alignment between $S[p...q]$ and $T[e...f]$. The score of the optimal *local* structural alignment between S and T can be obtained from the entry $\max_{e \leq f+1} A(1, m, e, f)$. We first show how to compute A, then show how to use the structure of S to guide the computation of A without considering all possible combinations of p, q.

When considering any substring $S' = S[x'...y']$ of $S[x...y]$, there are four possible cases: (1) S' is equal to S (i.e. $x' = x, y' = y$); (2) S' is a proper prefix in S (i.e. $x' = x, y' < y$); (3) S' is a proper suffix in S (i.e. $x' > x, y' = y$); (4) S' is a substring of $S[x + 1...y - 1]$ (i.e. $x' > x, y' < y$); Therefore, we can consider each case one by one when computing the value of A.

Define $A_1(p, q, e, f)$ be the score of the optimal *global* structural alignment between $S[p...q]$ and $T[e...f]$. Define $A_2(p, q, e, f)$ be the score of the optimal *prefix-global* structural alignment between $S[p...q-1]$ and $T[e...f]$. Define $A_3(p, q, e, f)$ be the score of the optimal *suffix-global* structural alignment between $S[p+1...q]$ and $T[e...f]$. Define $A_4(p, q, e, f)$ be the score of the optimal *semi-global* structural alignment between $S[p + 1...q - 1]$ and $T[e...f]$.

The value of $A(p, q, e, f)$ can be computed recursively and it is the maximum value of four cases: (1) when $S' = S[p, q]$ (i.e. $A_1(p, q, e, f)$); (2) when S' is a proper prefix of $S[p, q]$ (i.e. $A_2(p, q, e, f)$); (3) when S' is a proper suffix of $S[p, q]$ (i.e. $A_3(p, q, e, f)$; (4) when S' is a substring of $S[p + 1, q - 1]$ (i.e. $A_4(p, q, e, f)$; Lemma 1 summarizes these cases.

Lemma 1

$$A(p, q, e, f) = \max A_1(p, q, e, f), A_2(p, q, e, f), A_3(p, q, e, f), A_4(p, q, e, f)$$

The following subsections describe how to compute A_1, A_2, A_3, A_4.

5.1 Calculation of A_1

When considering the optimal global structural alignment (with affine gap model) between $S[p...q]$ and $T[e...f]$, there are nine possible cases: (1) $S[p]$ is aligned with $T[e]$ and $S[q]$ with $T[f]$; (2) $S[p]$ with $T[e]$ and $S[q]$ with space;(3) $S[p]$ with $T[e]$ and $T[f]$*withspace*; (4) $S[p]$ with space and $S[q]$ with $T[f]$; (5) $S[p]$ with space and $S[q]$ with space; (6) $S[p]$ with space and $T[f]$ with space; (7) $T[e]$ with space and $S[q]$ with $T[f]$; (8) $T[e]$ with space and $S[q]$ with space; (9) $T[e]$ with space and $T[f]$ with space. Hence, we can consider each case one by one when computing the value of A_1.

Define $A_{1x}(p, q, e, f)$, where $1 \leq x \leq 9$, be the score of the optimal *global* structural alignment between $S[p...q]$ and $T[e...f]$ where the above case x is satisfied. (i.e. if $x = 1$, then $S[p]$ is aligned with $T[e]$ and $S[q]$ with $T[f]$).

The value of $A_1(p, q, e, f)$ can be computed recursively and it is the maximum value of nine cases. Lemma 2 summarizes these cases.

Lemma 2

$$A_1(p, q, e, f) = \max \begin{cases} A_{11}(p, q, e, f), A_{12}(p, q, e, f), A_{13}(p, q, e, f), \\ A_{14}(p, q, e, f), A_{15}(p, q, e, f), A_{16}(p, q, e, f), \\ A_{17}(p, q, e, f), A_{18}(p, q, e, f), A_{19}(p, q, e, f), \end{cases}$$

We will describe the calculation of A_{12}. Similar skill can be applied for the others (i.e. A_{11}, A_{13}, ... , A_{19}).

Calculation of A_{12}. $A_{12}(p, q, e, f)$ is the score of the optimal global structural alignment between $S[p...q]$ and $T[e...f]$, which aligns $S[p]$ with $T[e]$ and $S[q]$ with space. There are three situations and we need to consider them one by one. Note that according to the affine gap model, the penalty of a first space in a gap (i.e. which is $h + s$) is different from the penalty of the other space in a gap (i.e. which is s). Situation I: when (p, q) is a base pair - aligning the base pair $S[p]$ with $T[e]$ and $S[q]$ with space. Considering the alignment between $S[p+1...q-1]$ and $T[e+1...f]$, if $S[q-1]$ is aligned with space (i.e. case 2, case 5 and case 8), then a penalty s should be considered. Otherwise (i.e. for the other six cases), a penalty $h+s$ should be considered. Situation II: when $\exists q'$ where $p < q' < q$ such that (p, q') is a base pair - we need to find $k \in [e-1, f]$ such that the sum of the alignment score between $S[p, q']$ and $T[e, k]$, and that between $S[q'+1, q]$ and $T[k+1, f]$ is maximum. Since $S[p]$ is aligned with $T[e]$ and $S[q]$ with space, the alignment between $S[p, q']$ and $T[e, k]$ should satisfy the case 1, case 2 and case 3 (i.e. $S[p]$ is aligned with $T[e]$). Similarly, the alignment between $S[q'+1, q]$ and $T[k+1, f]$ should satisfy the case 2, case 5 and case 8 (i.e. $S[q]$ is aligned with space). Situation III: when p does not form base pair with any base $q' \in [p, q]$ - we align base S[p] with T[e]. Then the alignment between $S[p+1...q]$ and $T[e+1...f]$ should satisfy the case 2, case 5 and case 8 (i.e. $S[q]$ is aligned with space). Lemma 3 summarizes these situations:

Lemma 3

$$A_{12}(p,q,e,f) = \max \begin{cases} //\textit{if } (p,q) \textit{ in} M_{p,q} \\ \max_{\substack{\alpha \in 11,13,14,16,17,19 \\ \beta \in 12,15,18}} \{A_\alpha[p+1, q-1, e+1, f] - h, \\ A_\beta[p+1, q-1, e+1, f]\} + \gamma(S[p], T[e]) - s \\ //\textit{if } \exists q' \textit{ where } p < q' < q \textit{ such that } (p,q') \textit{ is a base pair} \\ \max\limits_{\substack{e \le k \le f \\ \alpha \in \{11,12,13\} \\ \beta \in \{12,15,18\}}} A_\alpha[p, q', e, k] + A_\beta[q'+1, q, k+1, f] \\ //\textit{if } \nexists q' \textit{ such that } (p,q') \in M_{p,q} \\ \max_{\beta \in \{12,15,18\}} A_\beta[p+1, q, e+1, f] + \gamma(S[p], T[e]) \end{cases}$$

5.2 Calculation of A_2

When considering the optimal prefix-global structural alignment (with affine gap model) between $S[p...q]$ and $T[e...f]$, there are four possible cases: (1) $S[p]$ is aligned with $T[e]$; (2) $S[p]$ with space; (3) $T[f]$ with space; and (4) an empty string of S with T.

Define $A_{2x}(p, q, e, f)$, where $1 \le x \le 3$, be the score of the optimal *prefix-global* structural alignment between $S[p...q]$ and $T[e...f]$ where the above case x is satisfied. (i.e. if $x = 1$, then $S[p]$ is aligned with $T[e]$). Note that we do not need to define function for the case 4 because the corresponding score is

$-h-s(f-e+1)$. The value of $A_2(p,q,e,f)$ can be computed recursively and it is the maximum value of four cases. Lemma 4 summarizes these cases.

Lemma 4

$$A_2(p,q,e,f) = \max\{A_{21}[p,q,e,f], A_{22}[p,q,e,f], A_{23}[p,q,e,f], -h-s(f-e+1)\}$$

We will describe the calculation of A_{22}. Similar skill can be applied to calculate A_{21} and A_{23}.

Calculation of A_{22}. The following lemma lists out the computation of A_{22}.

Lemma 5

$$A_{22}(p,q,e,f) = \max \begin{cases} //\textit{if } (p,q) \textit{ in} M_{p,q} \\ \max_{\alpha\in 21,23} A_\alpha[p+1,q-1,e,f] - (h+s) \\ \max_{\alpha\in 22} A_\alpha[p+1,q-1,e,f] - (s) \\ \max_{\alpha\in 11,12,13,17,18,19} A_\alpha[p+1,q-1,e,f] - (h+s) \\ \max_{\alpha\in 14,15,16} A_\alpha[p+1,q-1,e,f] - (s) \\ //\textit{if } \exists q' \textit{ where } p<q'<q \textit{ such that } (p,q') \textit{ is a base pair} \\ \max\limits_{\substack{e-1\le k\le f \\ \alpha\in\{14,15,16\}}} A_\alpha[p,q',e,k] + A_2[q'+1,q,k+1,f] \\ A_{22}[p,q',e,f] \\ //\textit{if } \nexists q' \textit{ such that } (p,q') \in M_{p,q} \\ \max_{\alpha\in\{21,23\}} A_\alpha[p+1,q,e,f] - (h+s) \\ \max_{\alpha\in\{22\}} A_\alpha[p+1,q,e,f] - s \end{cases}$$

$A_{22}(p,q,e,f)$ is the score of the optimal prefix-global structural alignment between $S[p...q-1]$ and $T[e...f]$, where $S[p]$ is aligned with space. Similar to A_{12}, there are also the same three situations. Situation I: when (p,q) is a base pair - aligning the base pair $S[p]$ with space. Since a prefix of $S[p...q-1]$ is considered, there are two possibilities: a prefix of $S[p+1...q-1]$ is aligned with $T[e...f]$ (i.e. semi-global alignment), or the whole sequence $S[p+1...q-1]$ is aligned with $T[e...f]$ (i.e. global alignment). Situation II: when $\exists q'$ where $p<q'<q$ such that (p,q') is a base pair - we need to find $k\in[e-1,f]$ such that the sum of the alignment score between $S[p,q']$ and $T[e,k]$, and that between $S[q'+1,q]$ and $T[k+1,f]$ is maximum. Since a prefix of $S[p...q-1]$ is considered, there are two possibilities: (1) the whole sequence $S[p,q']$ is aligned with $T[e,k]$ (i.e. global alignment) and a prefix of $S[q'+1,q]$ is aligned with $T[k+1,f]$ (i.e. semi-global); (2) a prefix of $S[p,q']$ is aligned with $T[e,k]$ (i.e. semi-global) only. Situation III: when p does not form base pair with any base $q'\in[p,q]$ - we align base $S[p]$ with space. For each possibility of situation I & III, there are also two conditions: if $S[p+1]$ is aligned with $T[e]$ or $T[e]$ is aligned with space, the penalty score $h+s$ should be considered. Otherwise, if $S[p+1]$ is aligned with space, then the penalty score s should be considered. The lemma 5 summarizes these cases.

The calculations for A_3 and A_4 are similar. In the following subsection, we will describe the time complexity of the algorithm.

5.3 Time complexity

To fill the dynamic programming table, not all entries for all possible subrange of S needs to be filled. According to the design of the dynamic programming, there are three cases:

Case 1: If $(p, q) \in M_{p,q}$, then all the entries for $S[p, q]$ of all tables (i.e. A, A_1, A_2, A_3, A_4, A_{11}, ..., etc.) can be computed from the entries for $S[p-1, q+1]$.

Case 2: If $\exists q' < q$ s.t. $(p, q') \in M_{p,q}$, then all the entries for $S[p, q]$ of all tables can be computed from the entries for $S[p, q']$ and $S[q'+1, q]$.

Case 3: If $\nexists q'$ s.t. $(p, q') \in M_{p,q}$, then all the entries for $S[p, q]$ of all tables can be computed from the entries for $S[p+1, q]$.

Therefore, we define a function $\zeta(p, q)$ to determine for which set of subregions in S, we need to fill the corresponding entires in all the tables.

$$\zeta(p,q) = \begin{cases} \{(p+1, q-1)\} \text{ if } (p,q) \in M_{p,q} \\ \{(p+1, q'), (q'+1, q)\} \text{ if } \exists q' < q \text{ s.t. } (p,q') \in M_{p,q} \\ \{(p+1, q)\} \text{ if } \nexists q' \text{ s.t. } (p,q') \in M_{p,q} \end{cases}$$

We only need to fill in the entries for all the tables provided (p, q) can be obtained from $(1, m)$ by applying ζ function repeatedly. Considering the ζ function, each time the total size of the subregions outputted cannot be greater than the size of the input region and each of the subregions outputted is smaller than the input region. Therefore, in total there are only $O(m)$ such (p, q) values. Also, there are $O(n^2)$ values of different (e, f) values, and for each entry, it takes $O(n)$ because of the consideration of $e - 1 \leq k \leq f$ in the case that $\exists q' < q$ s.t. $(p, q') \in M_{p,q}$. After finishing the calculation of values $A(1, m, e, f)$ for all $1 \leq e, f \leq n$, the final answer (i.e. $\max_{e \leq f+1}\{A(1, m, e, f)\}$) can be computed in $O(n^2)$ time. Therefore the total time complexity $= O(mn^3) + O(n^2) = O(mn^3)$.

Theorem 1. *For any sequence $S[1..m]$ with regular structure and any sequence $T[1...n]$ with unknown structure, the optimal local alignment score between $S[1..m]$ and $T[1..n]$ can be computed in $O(mn^3)$.*

6 Conclusions

In the paper, we provided the first algorithm to handle local structural alignment with affine gap model of RNA with regular structure that compute the optimal alignment. Our experiments show that the solution is effective for ncRNA families in which members may have varying sizes on hairpins, loops or stems (contributing to large gaps) when compared to using only global alignment or local alignment without gap model. And also we have not yet studied different types of gap penalty model and the effect of setting different gap penalty parameters. A more detailed evaluation on our approach and comparison of it with other existing tools will be performed. Other interesting directions include speeding up the algorithm and considering other more complicated structures (e.g. the structures with pseudoknots).

Acknowledgements

The project is partially supported by the Seed Funding Programme for Basic Research (Project number: 200911159065) of the University of Hong Kong.

References

1. Frank, D.N., Pace, N.R.: Ribonuclease p: unity and diversity in a trna processing ribozyme. Annu. Rev. Biochem. 67, 153–180 (1998)
2. Nguyen, V.T., Kiss, T., Michels, A.A., Bensaude, O.: 7sk small nuclear rna blinds to and inhibits the activity of cdk9/cyclin t complexes. Nature 414, 322–325 (2001)
3. Wadler, C.S., Vanderpool, C.K.: A dual function for a bacterial small rna: Sgrs performs base pairing-dependent regulation and encodes a functional polypeptide. Proc. Natl. Acad. Sci. USA 104(51), 20454–20459 (2007)
4. Yang, Z., Zhu, Q., Luo, K., Zhou, Q.: The 7sk small nuclear rna inhibits the cdk9/cyclin t1 kinase to control transcription. Nature 414, 317–322 (2001)
5. Liu, C., Bai, B., Skogerbo, G., Cai, L., Deng, W., Zhang, Y., Bu, D., Zhao, Y., Chen, R.: Noncode: an integrated knowledge database of non-coding rnas. NAR 33(Database issue), D112–D115 (2005)
6. Griffiths-Jones, S., Bateman, A., Marshall, M., Khann, A., Eddy, S.R.: Rfam: an rna family database. NAR 31(1), 439–441 (2003), http://www.sanger.ac.uk/Software/Rfam/
7. Eddy, S.R.: Non-coding rna genes and the modern rna world. Nature Reviews in Genetics 2, 919–929 (2001)
8. Hertel, J., de Jong, D., Marz, M., Rose, D., Tafer, H., Tanzer, A., Schierwater, B., Stadler, P.F.: Non-coding rna annotation of the genome of trichoplax adhaerens. Nucleic Acids Research 37(5), 1602–1615 (2009)
9. Klein, R.J., Eddy, S.R.: Rsearch: Finding homologs of single structured rna sequences. BMC Bioinformatics 4(1), 44 (2003)
10. Zhang, S., Haas, B., Eskin, E., Bafna, V.: Searching genomes for noncoding rna using fastr. IEEE/ACM TCBB 2, 4 (2005)
11. Tabei, Y., Asai, K.: A local multiple alignment method for detection of non-coding rna sequences. Bioinformatics (2009) doi:10.1093/bioinformatics/btp261
12. Will, S., Reiche, K., Hofacker, I.L., Stadler, P.F., Backofen, R.: Inferring noncoding rna families and classes by means of genome-scale structure-based clustering. PLOS Computational Biology 3(4), e65 (2007)
13. Jiang, T., Lin, G., Ma, B., Zhang, K.: A general edit distance between rna structures. Journal of Computational Biology 9(2), 371–388 (2002)
14. Lin, G.H., Chen, Z.Z., Jiang, T., Wen, J.: The longest common subsequence problem for sequences with nested arc annotations. Journal of Computer and System Sciences 65(3), 465–480 (2002)
15. Nawrocki, E.P., Eddy, S.R.: Query-dependent banding (qdb) for faster rna similarity searches. PLoS Comput. Biol. 5, e56 (2007)
16. Le, S., Chen, J., Maizel, J.: Efficient searches for unusual folding regions in rna sequences. In: Structure and Methods: Human Genome Initiative and DNA Recombination, vol. 1, pp. 127–130. Adenine Pr (1990)
17. Rivas, E., Eddy, S.R.: Secondary structure alone is generally not statistically significant for the detection of noncoding rnas. Bioinformatics 16(7), 583–605 (2000)
18. Mosig, A., Zhu, L., Stadler, P.F.: Customized strategies for discovering distant ncrna homologs. Briefings in Functional Genomics and Proteomics (2009) (to be appear)

Fast Computation of the Exact Hybridization Number of Two Phylogenetic Trees

Yufeng Wu and Jiayin Wang

Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269, U.S.A.
{ywu,jiw09003}@engr.uconn.edu

Abstract. Hybridization is a reticulate evolutionary process. An established problem on hybridization is computing the minimum number of hybridization events, called the hybridization number, needed in the evolutionary history of two phylogenetic trees. This problem is known to be NP-hard. In this paper, we present a new practical method to compute the exact hybridization number. Our approach is based on an integer linear programming formulation. Simulation results on biological and simulated datasets show that our method (as implemented in program *SPRDist*) is more efficient and robust than an existing method.

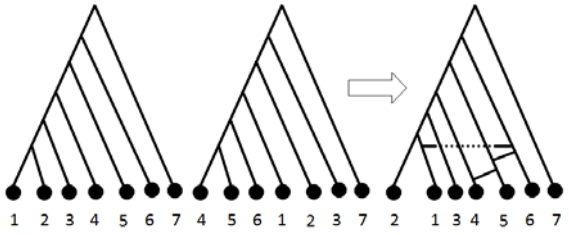
1 Introduction

Recently, *reticulate* evolutionary models have been actively studied in Phylogenetics. Several models have been proposed to address different reticulate processes (e.g. hybridization, lateral gene transfer and recombination). The literature on various aspects of reticulate evolution is growing rapidly. Refer to [12,11,18,15] for surveys of reticulate evolution. In this paper, we focus on *hybridization*. Hybridization refers to hybrid speciation, where hybrid species with mixed genetic composition from different species are created. Hybridization is believed to occur in many species [18].

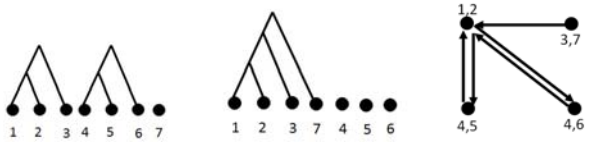
Imagine we have two phylogenetic trees (called gene trees), each for a gene of a (same) set of species. Due to reticulate evolution, the two gene trees are related but different. In this case, the evolutionary history of the two gene trees can not be represented by a single tree, but rather should be modeled as a *network*. This network is called "hybridization" network [5,18], which is closely related to the phylogenetic network or reticulate network [12,11,18,15]. Hybridization network for two gene trees is a directed acyclic graph, which is a compact representation of the two trees in the sense that it "displays" the two trees in a compact way. See Figure 1(a) for an example of hybridization networks.

Since there exist many feasible hybridization networks for two gene trees, a common formulation is to find the one with the *fewest* hybridization events. This is motivated by the belief that hybridization is relatively rare and thus the number of hybridization events is likely to be small. We call the *minimum* number of hybridization events in any hybridization network the *hybridization*

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 203–214, 2010.



(a) A hybridization network for T (left) and T' (middle)



(b) MAF (c) MAAF (d) G_{LP}

Fig. 1. (a): A hybridization network with three hybridization events. (b): maximum agreement forest of size three for T and T'. (c): maximum acyclic agreement forest of size four. (d): *part* of the leaf pair graph $G_{LP}(T, T')$.

number of the two gene trees. For a hybridization network of two gene trees, its hybridization number is equal to the number of nodes with in-degree two in the network. For example, the network in Figure 1(a) has three nodes with in-degree two, and thus its hybridization number is three. One should note that although hybridization number is only a quantity of the most parsimonious hybridization network, algorithms for computing the hybridization number can allow easy reconstruction of a parsimonious network itself as a by-product.

An established computational problem on hybridization networks, the *hybridization number problem*, is: given two trees, compute the hybridization number of the two trees. It is known that the hybridization number problem is NP-hard [8]. Nonetheless, computing the hybridization number of two trees has been studied in several papers [2,1,5,14]. A fundamental result in [1] showed that the hybridization number of two trees is equal to the size of the so-called Maximum Acyclic Agreement Forest (or MAAF) for the two trees minus one. See Section 2 for the definition of the MAAF formulation. There is a program, called *HybridNumber* [5], that can compute the exact hybridization number for two trees of moderate size or topologically similar. Program *HybridNumber* has worst-case exponential running time but is practical for certain range of data. The initial version of program *HybridNumber* was quite slow [5]. A later version of program *HybridNumber* appears to be much faster and more scalable [13]. However, our experience shows that program *HybridNumber* is still slow for larger data and also unstable in some cases: it crashed for some trees we tested due to software error. This greatly limits its application for larger biological data.

In this paper, we present a new method for computing the exact hybridization number of two gene trees based on an integer linear programming (ILP)

formulation. We also use a divide-and-conquer approach (developed in [5] and also used by program *HybridNumber*) in order to reduce the size of the problem. Our method is implemented in program *SPRDist*, which also outputs the corresponding maximum acyclic agreement forest and allows easy reconstruction of the most parsimonious hybridization network. To demonstrate the performance of our method, we provide simulation results on biological and simulated datasets. Comparing with program *HybridNumber*, our program *SPRDist* is more *efficient* for large trees, and also more *robust*.

2 Background

In this paper, we let a phylogenetic tree T be a binary *rooted* leaf-labeled tree. The set of the leaf labels of T is denoted as $L(T)$, its set of branches as $E(T)$, and its set of vertices as $V(T)$. Given a tree T, we can create a forest of trees $F(T) = \{T_1, T_2, \ldots, T_k\}$ from T by deleting a subset of $E(T)$. The forest $F(T)$ induces a partition of $L(T)$, and any two trees T_i and T_j are node disjoint. Conversely, we say a list of trees T_i form a forest for T if (a) for any tree T_i, $L(T_i) \subseteq L(T)$ and the union of $L(T_i)$ is equal to $L(T)$; (b) for each T_i, the (unique) *minimal* subtree connecting the nodes in $L(T_i)$, denoted as $S(T_i)$, is identical to T_i when nodes with degree two of $S(T_i)$ are contracted (called cleanup); if this holds, we say T_i *appears* in T; and (c) for any two trees T_i and T_j, $S(T_i)$ and $S(T_j)$ are node disjoint. The size of a forest is the number of trees in the forest.

Throughout this paper, we let two phylogenetic trees T and T' with the same set of leaf labels be the input data. Without loss of generality, we assume leaves are labeled with distinct integers from 1 to n, where n is the number of species. For convenience, we assign a distinct integer to each (leaf or internal) node in T and T', and thus we use the integer to refer to the node. When no confusion is caused, we assign integer i to a leaf labeled with i in both T and T'.

The concept of agreement forest has been used in many previous papers (e.g. [10,16,4,6]) and is also crucial to the current work. An agreement forest $F(T, T')$ is a set of trees $T_1, T_2, \ldots, T_k$ that is a forest for *both* T and T'. See Figure 1(b) for an illustration of agreement forest. Intuitively, an agreement forest is derived by cutting the same number of branches of T and T' which leads to the same set of trees (after cleanup). Agreement forest always exists for any two phylogenetic trees with the same set of leaves: a forest with n trees, each with a distinct leaf, is an agreement forest. We are interested in the agreement forest with the *smallest* number of trees (called maximum agreement forest or MAF). It is easy to see that the agreement forest in Figure 1(b) is also a MAF.

A useful observation [10,7] is that the rooted subtree prune and regraft (rSPR) distance between two trees is equal to the number of trees in a MAF minus one [10,7]. Recently, we developed an integer linear programming formulation (ILP), which can find the MAF of two trees and thus also compute the exact rSPR distance for many data [20]. Note that the two trees should be preprocessed to add a dummy leaf out of the root of each tree to ensure correctness, which we

perform throughout this paper. We now briefly describe the ILP formulation for the rSPR distance problem, since we will extend this formulation to the hybridization number problem.

Original ILP formulation for the MAF problem. The objective of the MAF problem is to find how to divide T and T' by cutting the *fewest* edges to derive an agreement forest. We define a binary variable C_i for each edge e_i in T (with m edges), where $C_i = 1$ if edge e_i is cut and 0 otherwise. We only place branch cuts in T and use T' as a reference. The objective of the ILP formulation is to minimize $\sum_{i=1}^{m} C_i$. We consider three leaves in both T and T'. We call the subtree connecting the three leaves in T the *triple* in T. Some triples do not agree in T and T'. For example, in Figure 1(a), the topology of triple of 1, 2 and 4 in T is different from that in T'. We call such triple "incompatible", meaning that this triple in T conveys different topological information from that in T'. To ensure the resulting forest to be an agreement forest, we require at least one edge of each incompatible triple is cut. Moreover, for two leaves i and j, we say the edges in T connecting i and j form a path between i and j. In Figure 1(a), the path between leaf pair $(1, 2)$ intersects that of leaf pair $(3, 4)$ in T' of Figure 1(a), but is disjoint in T. We call such two leaf pairs "incompatible", since the two paths for the two leaf pairs can not both be left uncut in an agreement forest. To ensure the resulting forest is an agreement forest, we require at least one edge along the two paths of two incompatible leaf pairs is cut. Both types of constraints can be easily expressed in ILP. The correctness of the ILP formulation for the MAF problem was established in [20]. See [20] for more details.

A useful observation on the hybridization number problem is made in [1], which concerns the so-called maximum *acyclic* agreement forest (MAAF). Maximum acyclic agreement forest is a special kind of agreement forest with one additional constraint, which concerns the topological order of the trees in the agreement forest. For agreement forest $F(T, T') = \{T_1, T_2, \ldots, T_k\}$, we say T_i is ancestral to T_j if the root of T_i is ancestral to the root of T_j in *either* T *or* T'. Here, a node v is ancestral to a node v' in T if v is on the (unique) path from the root to v' in T. Suppose we create a directed graph $G(F(T, T'))$ whose nodes are in one-to-one correspondence with $\{T_i\}$ (and thus we use T_i to refer both the node in the graph and the corresponding tree). To simplify notations, we write $G(F(T, T'))$ as $G(F)$. We create an edge from T_i to T_j if T_i in $G(F)$ is ancestral to T_j. An agreement forest $F(T, T') = \{T_1, T_2, \ldots, T_k\}$ is *acyclic* if the graph $G(F)$ is acyclic. As an example, the forest in Figure 1(b) is not acyclic, while the one in Figure 1(c) is. This is because in Figure 1(b), the tree with leaves 1, 2 and 3 is ancestral to the tree with leaves 4, 5 and 6 in T', and ancestral relationship is reversed in T. This leads to a cycle of two trees in $G(F)$. There are no cycles for the forest in Figure 1(c). We say a forest is maximum acyclic agreement forest (MAAF) if the forest is a MAF and acyclic. The forest in Figure 1(c) is a MAAF. The following theorem is proved in [1].

Theorem 1. *[1] The hybridization number of T and T' is equal to the size of the MAAF for T and T' minus one.*

3 Computing the Hybridization Number with ILP

Now we present an ILP formulation for finding the maximum acyclic agreement forest (MAAF) for T and T', which also allows the computation of the hybridization number due to Theorem 1. This ILP formulation also finds the MAAF, which can be used to reconstruct the most parsimonious hybridization network. We also apply a divide and conquer approach to speed up the computation.

3.1 New ILP Formulation for the MAAF Problem

Our ILP formulation is an extension of the ILP formulation for the MAF problem, as described in Section 2. We uses binary variable C_i to indicate whether edge e_i in T is cut or not. As in [20], we create binary variable $M_{i,j}$ for two (leaf or internal) nodes i and j in T. $M_{i,j} = 1$ iff *none* of the edges along the path between i and j is cut and 0 otherwise. Since a MAAF is also a MAF, we create the same set of ILP constraints as in the MAF formulation to ensure the resulting forest is an agreement forest. We now focus on how to ensure the resulting agreement forest $F(T, T')$ is acyclic in the ILP formulation.

Suppose the resulting forest $F(T, T')$ is known. Then it is straightforward to construct the corresponding graph $G(F)$ and test whether $G(F)$ is acyclic or not. A major problem here is that $G(F)$ depends on the agreement forest $F(T, T')$, which is exactly what we want to find. Without knowing $F(T, T')$, we can not explicitly construct $G(F)$. One may consider enumerating all possible agreement forests and then impose ILP constraints to forbid cycles. But enumerating agreement forests is impractical in most cases.

To get around this difficulty, our main idea is to consider *leaf pairs* in T. First note that the number of leaf pairs is much smaller than the number of possible agreement forests: there are $O(n^2)$ leaf pairs for n leaves. What is more important is that the acyclicity of $G(F)$ can be enforced using leaf pairs selected from the trees in $F(T, T')$ as we will explain later. Note that we do not know which leaf pairs i and j are in the same T_i without knowing $F(T, T')$. We do know, however, for each leaf pair of i and j, i and j are in the same tree of the forest iff $M_{i,j} = 1$. We now introduce the key tool of our approach: the leaf pair graph.

Leaf pair graph. We denote the leaf pair of two distinct leaves i and j as $lp(i, j)$. We say $lp(i, j)$ is connected if $M_{i,j} = 1$ (i.e. no branch is cut along the path between i and j) in T and also in T'. We say $lp(i, j)$ is from tree $T_i \in F(T, T')$ if $i, j \in L(T_i)$. Each connected leaf pair must be from some T_i of the forest. For two leaves i and j, we denote $MRCA_T(i, j)$ (respectively $MRCA_{T'}(i, j)$) as the most recent common ancestor of i and j in T (respectively T'). We say leaf pair $lp(i, j)$ is ancestral to leaf pair $lp(p, q)$ in T if $MRCA_T(i, j)$ is ancestral to $MRCA_T(p, q)$. We now construct the leaf pair graph $G_{LP}(T, T')$ (or simply G_{LP}), which is a directed graph. The nodes in G_{LP} are in one-to-one correspondence to the leaf pairs in T, and so we can use the leaf pairs to refer to the nodes of G_{LP}. For two leaf pairs $lp(i, j)$ and $lp(p, q)$ in G_{LP}, we create an edge from $lp(i, j)$ to $lp(p, q)$ if the following two conditions are *both* satisfied:

a. the path between i and j is disjoint with that of p and q in *both* T and T',
b. $lp(i,j)$ is ancestral to $lp(p,q)$ in *either* T *or* T'.

As an example, Figure 1(d) shows part of the leaf pair graph G_{LP} for the two trees in Figure 1(a). There is an edge from $lp(1,2)$ to $lp(4,5)$ because the MRCA of leaves 1 and 2 is ancestral to the MRCA of leaves 4 and 5 in T'. There is a degenerate case not covered by leaf pairs: leaf singletons in $F(T,T')$. But leaf singletons will not be part of any cycles in $G(F)$ because they can not be ancestral to any other trees: the root of a singleton tree is a leaf. Therefore, we only need to consider trees with at least two leaves from now on.

The reason that we require the two paths of two leaf pairs sharing an edge in G_{LP} to be disjoint is that we will only use one *realized* leaf pair per tree in the forest in our method. We say a leaf pair is realized in an agreement forest $F(T,T')$ if the two leaves are connected in $F(T,T')$. For a given agreement forest, some nodes (i.e. leaf pairs) of G_{LP} may not be realized. In this case, we remove from G_{LP} all leaf pairs that are not realized in $F(T,T')$ along with any edges incident to these leaf pairs. We denote the reduced G_{LP} as $G_{LP}(F)$, which is a sub-graph of G_{LP}. That is, $lp(i,j)$ appears in $G_{LP}(F)$ if i and j belong to the *same* tree in $F(T,T')$. We now show $G_{LP}(F)$ and $G(F)$ convey the same information on whether the agreement forest is acyclic or not.

Suppose $G_{LP}(F)$ contains a cycle, whose nodes are the (realized) leaf pairs. We first note that a cycle in $G_{LP}(F)$ can not contain only leaf pairs from a single tree of $F(T,T')$.

Lemma 1. *Cycles in* $G_{LP}(F)$ *contain leaf pairs from at least two trees in* $F(T,T')$.

Proof. For contradiction, suppose there exists a cycle in $G_{LP}(F)$ where all leaf pairs along the cycle are from a single tree in $F(T,T')$. Among these leaf pairs, we consider the leaf pair lp with the highest MRCA (i.e. closest to the root of the tree). Clearly, there exists no edge in G_{LP} that originates from some leaf pair on the cycle and points to lp. This contradicts the assumption that lp is on the cycle. □

Lemma 2. *For an agreement forest* F, $G(F)$ *is acyclic iff* $G_{LP}(F)$ *is acyclic.*

Proof. We will show that if $G(F)$ contains a cycle, then $G_{LP}(F)$ also contains a cycle, and vice versa. First suppose there is a cycle in $G(F)$ and suppose the cycle contains tree $T_{i_1},\ldots,T_{i_c}$. Then there exists one connected leaf pair from each T_{i_j} such that their MRCAs are the same as the *roots* of the trees which they belong to. These leaf pairs thus have the same ancestral relationship as T_{i_j}. Since the leaf pairs belong to different trees, the leaf pairs are pairwise disjoint. By the definition of $G_{LP}(F)$, these leaf pairs form a cycle in $G_{LP}(F)$.

Now suppose $G_{LP}(F)$ contains a cycle C. Due to Lemma 1, this cycle contains leaf pairs from at least two trees of $F(T,T')$. Note that each leaf pair on C must belong to some tree of $F(T,T')$, but there can be multiple leaf pairs belong to the same tree. We let $T'_1,T'_2,\ldots,T'_c$ be the *distinct* trees, which is ordered along C. Here $c \geq 2$ due to Lemma 1. We now show $T'_1,T'_2,\ldots,T'_c$ forms a cycle.

Consider two leaf pairs $lp(i,j)$ and $lp(p,q)$ that are consecutive along C. Moreover, they are from two different trees T'_i and T'_{i+1} respectively, and there is an edge from $lp(i,j)$ to $lp(p,q)$ in $G_{LP}(F)$. Then, in $G(F)$, there is an edge from T'_i to T'_{i+1}. To see this, without loss of generality assume $MRCA_T(i,j)$ is ancestral to $MRCA_T(p,q)$. Since T'_i and T'_{i+1} are disjoint, the root of T'_{i+1} must be on the path from $MRCA_T(i,j)$ to $MRCA_T(p,q)$ in T. Thus, there is a path from the root of T'_i to the root of T'_{i+1} in T, which leads to an edge from T'_i to T'_{i+1}. Since each T'_i has at least one leaf pair in C, there is an edge in $G(F)$ from each T'_i to T'_{i+1}. Therefore, trees $T'_1, \ldots, T'_c$ form a cycle in $G(F)$. □

Lemma 2 implies that if $G_{LP}(F)$ is acyclic, the resulting forest is acyclic. We will create ILP constraints on the leaf pairs that ensure acyclicity for the resulting $G_{LP}(F)$. Note that the proof of Lemma 2 suggests that we only need one leaf pair from a single tree of the forest.

If G_{LP} is acyclic, then *any* agreement forest created by some branch cuts will be acyclic. So we assume in the following that G_{LP} contains cycles. Imagine we enumerate all cycles in G_{LP}. If a leaf pair on a cycle is not realized by the resulting forest, the cycle will be absent from $G_{LP}(F)$. To ensure an agreement forest to be acyclic, we enforce for each cycle C, at least one node in C is not realized (i.e. the corresponding leaf pair is not connected in $F(T,T')$). Unfortunately, experimental study shows that G_{LP} can contain many cycles. This makes enumeration of cycles impractical. However, an *empirical* finding is that if we remove the so-called infeasible twin-pairs (defined later) from G_{LP}, then the reduced G_{LP} often contains a small number of cycles (and for many biological datasets, G_{LP} becomes acyclic).

Consider two leaf pairs $lp(i,j)$ and $lp(p,q)$, where there is an edge from $lp(i,j)$ to $lp(p,q)$ and an edge from $lp(p,q)$ to $lp(i,j)$ in G_{LP}. Here, $lp(i,j)$ and $lp(p,q)$ form a cycle of two nodes in G_{LP}. We call these two leaf pairs in G_{LP} *infeasible* twin-pairs. As an example, in Figure 1(d), $lp(1,2)$ and $lp(4,5)$ form an infeasible twin-pair, and so do $lp(1,2)$ and $lp(4,6)$. The two leaf pairs are called infeasible because to achieve an MAAF, at least one of the two leaf pairs is not realized. We now create ILP constraints to ensure at least one leaf pair of an infeasible twin-pair is not realized in $G_{LP}(F)$. We then remove from G_{LP} the two edges between two leaf pairs forming an infeasible twin-pair. This is valid since one of the two leaf pairs will not be realized and edges incident to both leaf pairs will always be removed.

In general, after deleting the edges between the infeasible twin-pairs, the reduced G_{LP} can still contain cycles. But our experience shows that for biological data, the reduced G_{LP} often contains a small number of cycles. This permits us to simply *enumerate* all possible elementary cycles in the reduced G_{LP}. A cycle is called elementary if it does not contain a smaller cycle. To enumerate elementary cycles in the directed graph G_{LP}, we use the algorithm developed by Tarjan [19], which appears to work well in our simulation study.

We now give the details on how our ILP formulation ensures the resulting forest to be acyclic. Recall that we have a binary variable $M_{i,j}$ for each pair of leaves i and j, where $M_{i,j} = 1$ if i and j are connected in the forest. We first

consider the infeasible twin-pairs in G_{LP}. For an infeasible twin-pair with $lp(i, j)$ and $lp(p, q)$, we impose an ILP constraint so that at least one of the leaf pairs is not realized:

$$M_{i,j} + M_{p,q} \leq 1$$

Second, we create one constraint for each enumerated elementary cycle C in the reduced G_{LP}. Suppose leaf pairs $(i_1, j_1), \ldots (i_c, j_c)$ are the nodes of C. Then we need to ensure at least one of these nodes is not realized:

$$\sum_{p=1}^{c} M_{i_p,j_p} \leq c - 1$$

These additional constraints are necessary and sufficient for the original MAF ILP formulation to obtain a MAAF, which we omit the detailed proof due to the space limit. To find the MAAF, we need to find which edges e_i with $C_i = 1$ are in the ILP solution. Empirical results show that our ILP formulation can often be solved efficiently in practice for many data (see Section 4).

3.2 Speed Up Computation by Divide and Conquer

For larger trees, it is known that computing the hybridization number can be made faster by preprocessing the input trees [5]. There are three preprocessing rules known to reduce the size of input trees while preserving the *optimality* of the solution. As implemented in program *HybridNumber*, these rules sometimes greatly reduce the running time. So we also use these rules by preprocessing the input trees before solving the ILP formulation.

The preprocessing applies to both T and T'. The first rule is: when there exists a common subtree T_0 for T and T' with at least two leaves, we delete T_0 from both T and T' and replace it with a single node $v(T_0)$. The reduced trees have the same hybridization number as the original trees [5]. The second rule is called *Chain Reduction* in [5], which is targeted to a special type of tree topology called maximal chain. Our experience suggests often this rule can not be applied to trees we tested. We refer the readers to [5] for more details.

The last rule, called cluster reduction in [5], is potentially more useful. Suppose there exists subtree T_1 of T, and also subtree T_2 of T', such that T_1 and T_2 may be topologically different but have the *same* set of leaves. Then we cut T_1 from T and T_2 from T' so that T_1 and T_2 become two new phylogenetic trees. We also add a new leaf, s, to T and T' at the same positions where T_1 and T_2 are previously attached. As shown in [5], the hybridization number of the original T and T' is equal to the summation of the hybridization number of the updated T and T' and that of T_1 and T_2. This rule can be effective because it can divide a larger problem into two smaller problems in a divide and conquer manner. However, not all input trees can be reduced by this rule.

To apply these preprocessing rules, we search for pairs of subtrees T_1 and T_2 with identical leaves in T and T'. Once the subtrees T_1 and T_2 are found, we cut

T_1 from T and T_2 from T', and then use the ILP approach described in Section 3.1 to compute the hybridization number of T_1 and T_2. We continue to divide the reduced T and T' until the two trees are small enough.

4 Results

We have implemented our method in an ILP based software tool called *SPRDist*. Program *SPRDist* was originally designed to compute the rSPR distance for two trees. We have updated program *SPRDist* to allow computation of the hybridization number for two trees. The tool is written in C++ and uses the GNU GLPK integer linear programming solver or the commercial CPLEX solver. Our experience shows CPLEX (a commercial ILP solver) is often faster and more robust than GLPK. To test the effectiveness of our method, we compute the hybridization number for a number of tree pairs from both simulated data and biological datasets. The experiment was performed on a 3192 MHz Intel Xeon workstation.

4.1 Simulated Data

The simulated data is from Beiko and Hamilton [3]. The trees are generated as follows: a random tree is first selected, and then a small number of random rSPR operations are applied to obtain the second tree. We tested ten pairs of trees, each with 100 leaves and the number of rSPR operations is equal to ten. In Table 1, we give results of these ten datasets. It can be seen that program SPRDist is efficient in computing the hybridization number of these trees: the running time is usually less than one minute. The CPLEX solver gives faster results for more difficult cases. Also the preprocessing helps to reduce the running time. As a comparison, program *HybridNumber* runs for very long time for most datasets: for only one of the ten datasets, program *HybridNumber* finds the solution within one hour.

4.2 Biological Data

To demonstrate that our method works for true biological data, we also test our method on the following biological data: tree pairs for a Poaceae dataset. The dataset is originally from the Grass Phylogeny Working Group [9]. The dataset contains sequences for six loci: internal transcribed spacer of ribosomal DNA (ITS); NADH dehydrogenase, subunit F (ndhF); phytochrome B (phyB); ribulose 1,5-biphosphate carboxylase/oxygenase, large subunit (rbcL); RNA polymerase II, subunit β'' (rpoC2); and granule bound starch synthase I (waxy). The Poaceae dataset was previously analyzed by Heiko Schmidt [17], who generated the inferred rooted binary trees for these loci. Bordewich, et al. [5] computed the minimum hybridization number for each of the fifteen pair of trees. Previously, we computed the exact rSPR distance for each pair of trees. To test how well our method performs on these biological trees, we compute the exact hybridization number for the same fifteen pairs of trees. See Table 2 for the results.

Table 1. Computing exact hybridization number on simulated data from Beiko and Hamilton (2006). For each pairs of trees, ten random rSPR operations are performed when the datasets are applied. h_S: the hybridization number computed by program *SPRDist*. Each pair is computed with CPLEX and GLPK. We also compare the program with or without divide-and-conquer preprocessing. The columns labeled as no prep. are for the results without performing preprocessing. Time is measured in seconds (s), hours (h) and days (d).

h_S	CPLEX	no prep.	GLPK	no prep.	HybridNumber
10	37 s	37 s	57 s	57 s	2 d 9 h
10	81 s	81 s	167 s	179 s	2 d 19 h
9	10 s	12 s	38 s	47 s	21 h 33 s
9	0 s	3 s	15 s	17 s	3205 s
10	15 s	16 s	66 s	82 s	2 d 6 h
9	0 s	5 s	1 s	26 s	2 d 20 h
10	13 s	13 s	53 s	58 s	2 d 7 h
10	18 s	24 s	318 s	379 s	2 d 8 h
10	16 s	376 s	72 s	146 s	2 d 5 h
10	3 s	10 s	29 s	77 s	2 d 9 h

Table 2. Performance of program SPRDist on fifteen pairs of trees for the Poaceae data. For comparison, we also list those of program *HybridNumber*. h_S: hybridization number from program SPRDist. rSPR: rSPR distance. h_{HN}: the minimum hybridization number computed in [5]. We list the running time of program *SPRDist* using either CPLEX and GLPK. We only give results for trees preprocessed with divide-and-conquer preprocessing. Time is measured in seconds (s) and hours (h).

	Data			SPRDist			HybridNumber	
Pair	1	2	# taxa	h_S(rSPR)	**CPLEX**	**GLPK**	h_{HN}	Time
1	ndhF	phyB	40	14 (12)	5 s	65 s	14	3 s
2	ndhF	rbcL	36	13 (10)	10 s	84 s	13	3 s
3	ndhF	rpoC2	34	12 (11)	7 s	77 s	12	6 s
4	ndhF	waxy	19	9 (7)	1 s	2 s	9	1 s
5	ndhF	ITS	46	19 (19)	**51** s	666 s	19	667 s
6	phyB	rbcL	21	4 (4)	0 s	1 s	4	1 s
7	phyB	rpoC2	21	7 (6)	3 s	2 s	7	1 s
8	phyB	waxy	14	3 (3)	1 s	1 s	3	1 s
9	phyB	ITS	30	8 (8)	1 s	2 s	8	1 s
10	rbcL	rpoC2	26	13 (11)	14 s	134 s	13	16 s
11	rbcL	waxy	12	7 (6)	1 s	1 s	7	1 s
12	rbcL	ITS	29	14 (13)	**80** s	1140 s	14	4 h 2716 s
13	rpoC2	waxy	10	1 (1)	0 s	0 s	1	1 s
14	rpoC2	ITS	31	15 (14)	**115** s	1469 s	15	7 h 776 s
15	waxy	ITS	15	8 (7)	1 s	9 s	8	2 s

The experimental results in Table 2 indicate that our program *SPRDist* is more efficient than program *HybridNumber* for more difficult datasets. In fact, our program only takes seconds or minutes to compute the *exact* hybridization number when CPLEX solver is used. The CPLEX solver usually leads to less running time than the GLPK solver, especially for more difficult input data. Even the GLPK solver can lead to faster running time than program *HybridNumber* for more difficult cases (pairs 12 and 14). On the other hand, program *HybridNumber* appears to perform better when the input trees allow significant reduction through preprocessing; when preprocessing is less effective, it usually performs poorly for larger trees. This can also be seen in Table 1. We also compare the hybridization number with the rSPR distance. It appears that the two values tend to be more different when the size of trees and the hybridization number grow.

The simulation results show that our program *SPRDist* is more scalable than program *HybridNumber*. Also, program *SPRDist* is more robust than program *HybridNumber*. With our new method, computing the hybridization number between two large and topologically far apart trees is still challenging, but feasible.

Constructing the consistent history. In addition to computing the hybridization number in this paper, our method also finds the corresponding maximum acyclic agreement forest. It is straightforward to apply the algorithm given in [18] to construct a most parsimonious phylogenetic history from the forest.

Acknowledgment. Research is supported by National Science Foundation [IIS-0803440] and the Research Foundation of University of Connecticut.

References

1. Baroni, M., Grunewald, S., Moulton, V., Semple, C.: Bounding the number of hybridization events for a consistent evolutionary history. J. Math. Biol. 51, 171–182 (2005)
2. Baroni, M., Semple, C., Steel, M.: A framework for representing reticulate evolution. Annals of Combinatorics 8, 391–408 (2004)
3. Beiko, R.G., Hamilton, N.: Phylogenetic identification of lateral genetic transfer events. BMC Evolutionary Biology 6, 15 (2006)
4. Bonet, M., John, K.S., Mahindru, R., Amenta, N.: Approximating subtree distances between phylogenies. J. of Comp. Biology 13, 1419–1434 (2006)
5. Bordewich, M., Linz, S., John, K.S., Semple, C.: A reduction algorithm for computing the hybridization number of two trees. Evolutionary Bioinformatics 3, 86–98 (2007)
6. Bordewich, M., McCartin, C., Semple, C.: A 3-approximation algorithm for the subtree distance between phylogenies. J. Discrete Algorithms 6, 458–471 (2008)
7. Bordewich, M., Semple, C.: On the computational complexity of the rooted subtree prune and regraft distance. Annals of Combinatorics 8, 409–423 (2004)
8. Bordewich, M., Semple, C.: Computing the minimum number of hybridization events for a consistent evolutionary history. Discrete Applied Mathematics 155, 914–928 (2007)

9. Grass Phylogeny Working Group. Phylogeny and subfamilial classification of the grasses (poaceae). Ann. Mo. Bot. Gard. 88, 373–457 (2001)
10. Hein, J., Jiang, T., Wang, L., Zhang, K.: On the complexity of comparing evolutionary trees. Discrete Appl. Math 71, 153–169 (1996)
11. Huson, D., Bryant, D.: Application of phylogenetic networks in evolutionary studies. Molecular Biology and Evolution 23, 254–267 (2006)
12. Linder, C.R., Moret, B.M.E., Nakhleh, L., Warnow, T.: Network (reticulate) evolution: biology, models, and algorithms (2004)
13. Linz, S.: Personal communications
14. Linz, S., Semple, C.: Hybridization in nonbinary trees. IEEE/ACM Transactions on Computational Biology and Bioinformatics 6, 30–45 (2009)
15. Nakhleh, L.: Evolutionary phylogenetic networks: models and issues. In: Heath, L., Ramakrishnan, N. (eds.) The Problem Solving Handbook for Computational Biology and Bioinformatics. Springer, Heidelberg (In press 2010)
16. Rodrigues, E.M., Sagot, M.F., Wakabayashi, Y.: Some approximation results for the maximum agreement forest problem. In: Proceedings of RANDOM-APPROX 2001, pp. 159–169 (2001)
17. Schmidt, H.: Phylogenetic trees from large datasets. PhD thesis, Heinrich-Heine-Universität, Düsseldorf (2003)
18. Semple, C.: Hybridization networks. In: Gascuel, O., Steel, M. (eds.) Reconstructing Evolution: New Mathematical and Computational Advances, Oxford, pp. 277–309 (2007)
19. Tarjan, R.: Enumeration of the elementary circuits of a directed graph. SIAM J. on Computing 2, 211–216 (1973)
20. Wu, Y.: A practical method for exact computation of subtree prune and regraft distance. Bioinformatics 25, 190–196 (2009)

"Master-Slave" Biological Network Alignment

Nicola Ferraro[1], Luigi Palopoli[1], Simona Panni[2], and Simona E. Rombo[1]

[1] DEIS, Università della Calabria
[2] Dept. of Cellular Biology, Università della Calabria

Abstract. Performing global alignment between protein-protein interaction (PPI) networks of different organisms is important to infer knowledge about conservation across species. Known methods that perform this task operate symmetrically, that is to say, they do not assign a distinct role to the input PPI networks. However, in most cases, the input networks are indeed distinguishable on the basis of how well the corresponding organism is biologically well-characterized. For well-characterized organisms the associated PPI network supposedly encode in a sound manner all the information about their proteins and associated interactions, which is far from being the case for not well characterized ones. Here the new idea is developed to devise a method for global alignment of PPI networks that in fact exploit differences in the characterization of organisms at hand. We assume that the PPI network (called *Master*) of the best characterized is used as a fingerprint to guide the alignment process to the second input network (called *Slave*), so that generated results preferably retain the structural characteristics of the Master (and using the Slave) network. We tested our method showing that the results it returns are biologically relevant.

1 Introduction

High-throughput technologies, including genome sequencing, expression profiling, cellular localization and other methods for large-scale protein-protein interactions, have provided a large amount of information for few well-characterized model organisms such, as for instance, the yeast *Saccharomyces cerevisiae* [6, 11]. On the other hand, for many organisms, the genome sequence has been determined, but coding sequences have been functionally annotated on the sole basis of sequence similarity. Although it is certainly true that similar protein sequence implies similar protein function, inferring protein functions of not yet well characterized organisms by exploiting protein sequence similarity to other organism proteins may be complicated by specie-specific diversifications or when species are not closely related. Furthermore, it has been noted that to fully understand cell activity, proteins cannot be analyzed independently from the other proteins of the same organism, because they seldom act in isolation to perform their tasks [18].

The protein-protein interactions of a given organism are usually modeled by a network, called *protein-protein interaction (PPI) network*, highlighting the mutual interactions between pairs of proteins. By comparing the PPI networks of different organisms the complex mechanisms at the basis of evolutionary conservations can be uncovered and the biological meaning of groups of interacting proteins belonging to not yet well

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 215–229, 2010.

characterized organisms can be thus inferred. As a result, a number of approaches have been recently presented in the literature for local [13, 3] and global [15, 16, 7, 10, 12] alignment of PPI networks.

In this context, the research presented here deals with global alignment of PPI networks. Global network alignment aims at finding a unique (possibly, the best) overall alignment of the input networks, in such a way that all the nodes of the networks are mapped. Unfortunately, exact algorithms for PPI network global alignment cannot be afforded, inasmuch as the PPI network alignment problem can be reduced to subgraph isomorphism checking, that is known to be NP-complete [5] and, therefore, heuristic approached are to be adopted.

A common characteristics of known methods for global alignment handle their input PPI networks symmetrically, that is to say, they do not take advantage of the (usually available) knowledge about how well the corresponding organisms are biologically well-characterized. Indeed, while for well-characterized organisms, the associated PPI networks supposedly encode in a sound manner all the information about their proteins and associated interactions, this is far from being the case for not well characterized ones. Therefore, it seems sensible to devise methods for global alignment that in fact exploit differences in the characterization of the organisms at hand, which is precisely the main idea underlying this paper. In particular, in our approach, the PPI network (called *Master*) of the best characterized organism is used as a fingerprint to guide the alignment process to the second input network (called *Slave*), so that generated results preferably retain the structural characteristics of the Master network. This is obtained by generating from the Master (and using the Slave) a finite automaton, called *alignment model*, which is then fed with a (linearization of) the Slave network for the purpose of generating, via the Viterbi algorithm, matching subgraphs. In this way most of the structural information of the Master is kept, while only the Slave information useful to understand how much of the Master has been conserved in the Slave is exploited in the alignment process. Such an asymmetric alignment may be relevant for example when the Master network is refined with information taken from multiple literarure sources, also taking into account the accuracy of each reported interaction (see [9] for the *Saccaromices cerevisiae* network). Indeed, the Master may contain in this case valuable information for the search of known complexes modelling the cell machinery of other less studied organisms.

While our technique is valuable in all those cases where the biological characterization of the input organisms is rather different, it can demonstrate itself useful also in cases where the two input networks are in fact equally well characterized. Indeed, in such cases, one of the input networks can be set as the Master, while the other is used as the Slave, thus "constraining" the alignment process to be preferably bound to the first network structural characteristics. The process can be then continued by exchanging the roles of the two networks at hand.

In more detail, the technique presented here amounts to iteratively extracting similar connected subgraphs from the input networks. The algorithm starts by searching for an initial seed, that is, a *best pair* of proteins (p, q) (one from the first network and one from the second) to be matched. To this end, information about both protein sequence

similarity and networks topology are used[1]. Then, the seed is expanded to a pair of matching subgraphs of the two input networks by exploring the nodes adjacent to p and q. When a new pair of connected subgraphs is eventually discovered, the two subgraphs are deleted from the input networks and the subgraph extraction procedure is started again. The process is iterated until no further solutions can be generated. The set of all the protein pairings resulting from the discovered subgraph matchings makes the global alignment between the input networks.

In order to asses the effectiveness of the approach, several experiments have been conducted over the PPI networks of *Saccharomyces cerevisiae* (yeast) and *Drosophila melanogaster* (fly). Experimental results on the these two networks demonstrate that our technique is able to find biologically significant subgraph pairings, some of which are not generated by other global alignment methods.

The rest of the paper is organized as follows. In Section 2 some basics concepts are illustrated. In Section 3 the procedure to match connected subgraphs is described, while Section 4 illustrates the algorithm we propose to perform global alignment of two PPI networks. In Section 5, the main results obtained by applying the technique to align the yeast and fly networks are illustrated. Finally, in Section 6, brief conclusions are drawn.

2 Preliminaries

A PPI network can be modeled as an indirect graph $\mathcal{N} = \langle P, I \rangle$, where P is a set of nodes, each denoting a specific protein in the considered organism, and I is the set of edges representing protein-protein interactions. Nodes can be labeled by protein names or by database ids. Now, let us denote with $a \circ b$ the concatenation of elements (or pairs) a and b. Analogously, for elements (or pairs) $a_1, a_2, \ldots, a_n$, $\circ_{1,n} a_i$ denotes $(a_1 \circ (a_2 \circ (\ldots \circ (a_{n-1} \circ a_n))))$, and, for an ordered set A, $\circ_{a_i \in A} a_i$ denotes $(a_1 \circ (a_2 \circ (\ldots \circ (a_{n-1} \circ a_n))))$ where $A = \langle a_1, a_2, \ldots, a_n \rangle$. Furthermore, given a PPI network $\mathcal{N} = \langle P, I \rangle$ and a node $p \in P$, the *adjacency set* of p is the set $adj(p) = \{q \in P | \{p, q\} \in I\}$ of nodes adjacent to p.

Next we introduce the technical machinery useful to our purposes. We begin by modeling the Master network by defining its associated automata, called the *alignment model*, which is defined below.

Definition 1. *(Alignment model)* Let $\mathcal{N}_M = \langle P_M, I_M \rangle$ and $\mathcal{N}_S = \langle P_S, I_S \rangle$ be two PPI networks that we call *Master* and *Slave*, resp., and let k be an integer such that $k \geq 1$. Furthermore, let D be a set of triplets $\langle p, q, s_{pq} \rangle$ and s_{th} be a real value such that for $p \in P_M$ and $q \in P_S$, s_{pq} is the similarity value for p and q and s_{th} is a threshold value. Finally, let v and v' be two values such that $v < v'$.

An *alignment model M of order k* for $\mathcal{N}_M$ w.r.t. $\mathcal{N}_S$ is a finite state automaton such that:

- the states of the automaton include one state for each protein in $P_M \cup P_S$ and, moreover, states β, τ, and a set of states ϵ_h defined as follows;
- β is the initial state and it is linked to itself by a transition with value v;

[1] Other kinds of information about protein structure might be taken advantage of as well.

- the state τ is linked to itself by a transition with value v;
- each node of P_M corresponds to a state of level 0, presenting an input transition from the node β with value v', and an output transition towards the node τ with value v;
- for each state of level $i = 0, \ldots, k-2$ corresponding to a node $p \in P_M$, there is a set of states of level $i+1$ linked in input and in output to the state of level i by transitions with value v'. Each state of level $i+1$ corresponds to a node $p' \in adj(p)$;
- each state of level $i = 0, \ldots, k-2$ corresponding to a node $p \in P_M$, is linked to a state ϵ_{i+1} by a transition with value v. The state ϵ_{i+1}, in its turn, is linked to itself and to the node p by transitions with value v;
- states β and τ emit any symbol with emission value equal to 1;
- each state of level i corresponding to a node $p \in P_M$ emits symbols of the type (q, i) $(q \in P_S)$ whose emission value is equal to 1 if $s_{pq} \geq s_{th}$, while it is equal to 0 otherwise;
- each state ϵ_i emits symbols of the type (q, j) $(q \in P_S, j \geq i)$ with emission value 1, and all the other symbols with emission value 0.

Figure 1 shows the generic structure of an alignment model of order two.

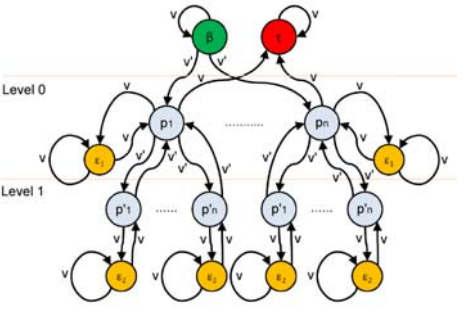


Fig. 1. An alignment model of order two. Nodes represent the states of the automaton, edges represent the transitions.

Note that each output sequence of an alignment model can be obtained by following the different paths in the model. Each path has a specific value, and goes through at most one state of level 0, even several times. Furthermore, the value v' characterizes input/output transitions to/from states corresponding to nodes in P_M, while the value v characterizes transitions corresponding to the other states.

Let π be a path of the alignment model and $w(\pi)$ be its *weight*, that is, the sum of the values of the transitions in π. Intuitively, we point out that paths scoring high weights will correspond to good pairings between Master and Slave nodes, as will be more clear below. Indeed, the weights give a measure of how much the Slave "matches" the Master for the nodes involved in the corresponding paths.

As already pointed out in the Introduction, in order to perform the alignment process using the alignment model, the Slave network has to be first linearized. The following definitions are useful to this aim.

Definition 2. *(k-tour)* Let $\mathcal{N} = \langle P, I\rangle$ be a PPI network, $p \in P$ and k be an integer, $k \geq 1$. A k-tour for p, is defined as $tour_k(p) = \langle T_k(p, 0)\rangle$ where, for a generic node a:

- $T_k(a, k-1) = (a, k-1)$,
- $T_k(a, i) = (a, i) \circ (\circ_{b \in adj(a)}(T_k(b, i+1) \circ (a, i))), \forall i < k-1$.

Example 1. Consider the graph illustrated in Figure 2.

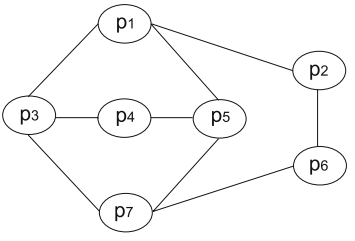


Fig. 2. A sample graph

For the node p_5, we have the following k-tours (for $k = 1, 2, 3$):

- $tour_1(p_5) = \{(p_5, 0)\}$
- $tour_2(p_5) = \{(p_5, 0), (p_4, 1), (p_5, 0), (p_7, 1), (p_5, 0), (p_1, 1), (p_5, 0)\}$
- $tour_3(p_5) = \{(p_5, 0), (p_4, 1), (p_3, 2), (p_4, 1), (p_5, 2), (p_4, 1), (p_5, 0),$
  $(p_7, 1), (p_3, 2), (p_7, 1), (p_5, 2), (p_7, 1), (p_6, 2), (p_7, 1), (p_5, 0),$
  $(p_1, 1), (p_3, 2), (p_1, 1), (p_5, 2), (p_1, 1), (p_2, 2), (p_1, 1), (p_5, 0)\}$

The following definition extends previous Definition 2 to leave out a specific group of nodes from the adjacent sets under consideration.

Definition 3. *(partial k-tour)* Let $\mathcal{N} = \langle P, I\rangle$ be a PPI network, $p \in P$, k be an integer, $k \geq 1$ and Q be a subset of P. A partial k-tour for p is defined as:

$$ctour_k(p, Q) = \langle T_k(p, 0, Q)\rangle,$$

where, for a generic node a:

- $T_k(a, k-1, Q) = (a, k-1)$,
- $T_k(a, i, Q) = (a, i) \circ (\circ_{b \in adj(a)-Q}(T_k(b, i+1, Q) \circ (a, i))), \forall i < k-1$.

Both a k-tour and a partial k-tour can be referred to a specific set of nodes $Q' \subseteq P$. In such a case, they are denoted by $tour_k(Q') = \{\circ_{p \in ord(Q')} tour_k(p)\}$ and $ctour_k(Q', Q) = \{\circ_{p \in ord(Q')} ctour_k(p, Q)\}$, respectively, where $ord(Q')$ is any given permutation of the elements of Q'.[2]

[2] Depending on the chosen permutation, different tours are generated, but this choice is immaterial for our purposes.

3 Subgraph Extraction

This section describes the technique designed to match connected subgraphs. As already pointed out, it uses alignment models and k-tours defined in Section 2, along with the Viterbi algorithm [4]. The Viterbi algorithm has been proposed in 1967 [17] as a method of decoding convolutional codes, and it has been also exploited to solve the problem of estimating the state sequence of a discrete-time finite-state Markov process observed in memoryless noise [4]. In this work, we apply it to find the path scoring the maximum weight on the alignment model, without referring to any probabilistic meaning. In the following, we assume that, for each pair of proteins belonging to distinct networks, a *basic* similarity value (e.g., protein sequence similarity [1]) is known and stored in a suitable dictionary.

Let $\mathcal{N}_M = \langle P_M, I_M \rangle$ and $\mathcal{N}_S = \langle P_S, I_S \rangle$ be the two input PPI networks, where $\mathcal{N}_M$ is the Master and $\mathcal{N}_S$ is the Slave. Let D be a *dictionary of basic similarities*, that is, a set of triplets $\langle p_x, p_y, s_b \rangle$ such that $p_x \in P_M$, $p_y \in P_S$ and s_b is the *basic similarity* between p_x and p_y. Finally, let k be an integer such that $k \geq 1$. The procedure *Connected-subgraphs Extraction* includes two main steps:

1. find the pair of nodes (p_0, q_0), such that $p_0 \in P_M$ and $q_0 \in P_S$, to be set as *best-pair*, that is, the seed pair of nodes making the starting solution S_0;
2. expand S_0 to obtain the solution S_f corresponding to a pair of similar connected subgraphs C_L and C_F of the two input networks.

Step 1 and Step 2 are performed by two algorithms, called *Best-pair Finder* and *Expander*, that are described in detail in the following sections.

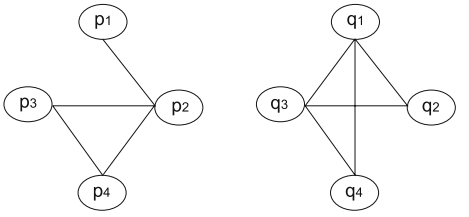
3.1 Best-Pair Finder

Given the two networks $\mathcal{N}_M$ and $\mathcal{N}_S$, the integer k and the dictionary D of basic similarity in input, *Best-pair Finder* returns in output the best-pair (p_0, q_0) as follows.

An alignment model M of order k for $\mathcal{N}_M$ w.r.t. $\mathcal{N}_S$ is generated, and a k-tour T_F for the set of nodes in $\mathcal{N}_S$ is considered as the output sequence of M. Here, high weights of the paths on M correspond to good pairings between Master and Slave nodes. In fact, the value $w(\pi)$ of a path π gives a measure of how much the Master node corresponding to the state of level 0 in the path "matches" with the emitted symbol, that corresponds to a Slave node. The notion of "good matching" we adopt is referred to the basic similarity associated with both p_0 and q_0, and their correspondent adjacent nodes.

To obtain the best match between a node p_0 of the Master and a node q_0 of the Slave, the path π scoring the maximum weight has to be chosen, and the Viterbi algorithm [4, 8] is exploited to this aim.

Example 2. Consider the two networks shown in Figure 3 (a), where the lef-most one is the Master and the right-most one is the Slave. We set $k = 2$ and in Figure 3 (b) the pairs of proteins whose basic similarity is greater than the input threshold are shown. Figure 4 illustrates the alignment model M of order k for the Master w.r.t. the Slave (we adopted a compact view in which the transition values v and v' are omitted and the ϵ states are represented by circles adjacent to the corresponding nodes).



(a)

p_1	q_1
p_2	q_4
p_3	q_3
p_4	q_1

(b)

Fig. 3. (a) Master and Slave networks (b) Proteins pairs scoring basic similarities greater than the threshold

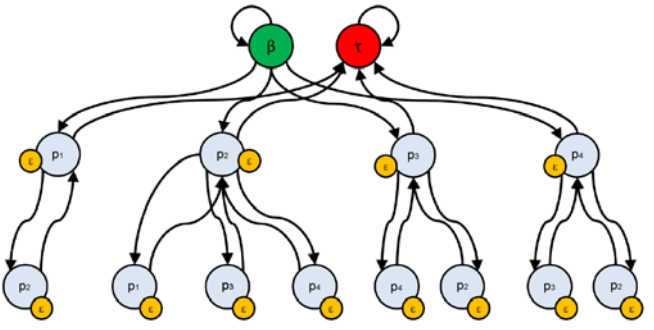


Fig. 4. The alignment model of order 2 for the Master w.r.t. to the Slave

Consider the following 2-tour as one of the possible output sequences of M:

$$T_F = \{tour_2(q_1), tour_2(q_3), tour_2(q_2), tour_2(q_4)\} =$$

$$\{(q_1,0),(q_3,1),(q_1,0),(q_4,1),(q_1,0),(q_2,1),(q_1,0),$$
$$(q_3,0),(q_1,1),(q_3,0),(q_4,1),(q_3,0),(q_2,1),(q_3,0),$$
$$(q_2,0),(q_1,1),(q_2,0),(q_3,1),(q_2,0),$$
$$(q_4,0),(q_3,1),(q_4,0),(q_1,1),(q_4,0)\}.$$

When the Viterbi algorithm is applied, the following path on M is returned:

$$\pi = \beta, p_4, p_3, p_4, p_2, p_4, \epsilon, p_4, \tau, \ldots, \tau.$$

The path π associates the first symbol of T_F, that is, the node q_1 of the Slave, to the Master node corresponding to the state of level 0 in π, that is, p_4. Therefore, the returned best-pair is (p_4, q_1).

However, to better understand why this is the returned solution, let us consider the following five alternative paths on M:

$$\pi_1 = \beta, p_1, \epsilon, p_1, p_2, p_1, \epsilon, p_1, \tau, \ldots, \tau,$$

$$\pi_2 = \beta, \ldots, \beta, p_3, p_4, p_3, p_2, p_3, \epsilon, p_3, \tau, \ldots, \tau,$$

$$\pi_3 = \beta, \ldots, \beta,$$
$$\pi_4 = \beta, \ldots, \beta, p_2, p_3, p_2, p_1, p_2, \tau,$$
$$\pi_5 = \beta, \ldots, \beta, p_2, p_3, p_2, p_4, p_2, \tau.$$

Following the same reasoning as before, the path π_1 leads to the pairing of nodes p_1 and q_1. Note that π_1 passes through three v' transitions, whereas π passes through five v' transitions. Since q_1 has associated a sufficiently high basic similarity with both p_1 and p_4, pairing p_4 and q_1 produces a better matching than pairing p_1 and q_1. Note that such a better matching depends on both the topology and the node similarities characterizing the two networks.

Both the paths π_4 and π_5 produce the pairing (p_2, q_4), while the path π_2 pairs p_3 and q_3. Note that, like π, the paths π_2, π_4 and π_5 also pass through five v' transitions. Thus, each one of the pairs (p_4, q_1), (p_3, q_3) and (p_2, q_4) would produce a good matching.

Finally, we note that path π_3 is a special one. In fact, it does not contain any state of level 0, and does not pass through any state characterized by v' transitions. Thus, it would be returned by the Viterbi algorithm only if there is no pair of proteins sharing basic similarities greater than the threshold.

3.2 Expander

Once that the best-pair of proteins composing the starting solution $S_0 = \{(p_0, q_0)\}$ is computed by *Best-pair Finder*, S_0 has to be expanded until no more proteins belonging to connected sub-graphs we are generating can be paired.

The *Expander* takes in input two networks $\mathcal{N}_M$ and $\mathcal{N}_S$, an integer k, the current solution S_0 and the basic similarity dictionary D, and returns in output the solution S_f corresponding to matching two connected subgraphs in the input networks.

To expand S_0, the *Expander* algorithm first analyzes the adjacent sets $adj(p_0)$ and $adj(q_0)$ to find a suitable pair (p_1, q_1), such that $p_1 \in adj(p_0)$ and $q_1 \in adj(q_0)$, to be added to S_0. This process leads to the generation of a new partial solution $S_1 = \{(p_0, q_0), (p_1, q_1)\}$. The algorithm works analogously to expand S_1 until the final solution S_f is generated.

At the generic step i, the pair (p_i, q_i) is computed according to the following procedure. Let $S_{i-1} = \{(p_0, q_0), (p_1, q_1), (p_2, q_2), \ldots, (p_{i-1}, q_{i-1})\}$ be the solution at the step $i - 1$. A partial k-tour T_{F_p} for the set of nodes in $\mathcal{N}_S$ on the set $Q = \{q_0, q_1, \ldots, q_{i-1}\}$ is generated, as well as a special alignment model M_p for $\mathcal{N}_M$. This model is obtained accordingly to the following variant of Definition 1:

- Nodes in the set $P = \{p_0, p_1, p_2, \ldots, p_{i-1}\}$ can not be associated to states of level greater than 0, and only nodes in P are states of level 0. Furthermore, states of level 0 are not linked to any state ϵ, nor to the state τ;
- each state of level 0 emits symbols of the type (p, q), such that $p \in P$ and $q \in Q$ with value 1, and any other symbol with value 0;
- there is a transition with value v from each node of level 1 to the node τ;
- there are no transitions from nodes of level 1 to nodes of level 0;

We call *partial alignment model* the alignment model M_p generated as described above. Differently from the alignment model of Definition 1, M_p allows to select pairs of proteins belonging to the adjacent sets of already chosen proteins, obtaining the correspondence between connected subgraphs as a final solution.

The partial tour T_{F_p} is used as the output sequence of M_p, and the Viterbi Algorithm is applied again to find the path π scoring maximum weight. Then, the pair (p_i, q_i) corresponding to π is added to S_{i-1}, generating this way the new solution S_i.

Note that, in the partial alignment model, only nodes of level 1 concur to generate the solution, while nodes of level 0 guarantee that, if the subgraphs generated at the previous step are connected, the new ones will be connected as well, and sharing the same spanning tree.

Example 3. Consider again the two networks in Figure 3. As discussed in Example 2, *Best-pair Finder* returns the solution $S_0 = \{(p_4, q_1)\}$. Consider the following partial 2-tour for the set $Q = \{q_1\}$:

$$T'_{F_p} = \{(q_1, 0), (q_3, 1), (q_1, 0), (q_4, 1), (q_1, 0), (q_2, 1), (q_1, 0)\}.$$

Figure 5 (a) shows the partial model M'_p built w.r.t. the set $P = \{p_4\}$.

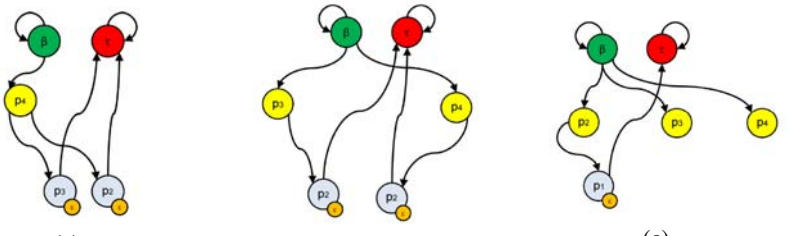


Fig. 5. (a) The partial alignment model M'_p (b) The partial alignment model M''_p (c) The partial alignment model M'''_p

When *Expander* is called, the path $\pi' = \beta, p_4, p_3, \tau, \ldots, \tau$ is returned by the Viterbi algorithm and the solution $S_1 = \{(p_4, q_1), (p_3, q_3)\}$ is generated at the first iteration. Then, the partial tour $T''_{F_p} = \{(q_1, 0), (q_1, 0), (q_4, 1), (q_1, 0), (q_2, 1), (q_1, 0)\}$ and the partial model M''_p displayed in Figure 5 (b) are produced, leading to the solution $S_2 = \{(p_4, q_1), (p_3, q_3), (p_2, q_4)\}$.

At the third iteration, the partial tour $T'''_{F_p} = \{(q_1, 0), (q_2, 1), (q_1, 0), (q_3, 0), (q_2, 1), (q_3, 0), (q_4, 0)\}$ and the partial model M'''_p shown in Figure 5 (c) are generated. In this case, the Viterbi algorithm returns the path $\pi''' = \{\beta, \beta, \beta, \beta, \beta, \beta, p_2\}$.

Since π''' does not contain any state of level 1, this means that no further node can be added to the final solution, that is then S_2. The constructed match between the two connected subgraphs is shown in Figure 6.

4 Global Alignment

To perform a global alignment between two networks $\mathcal{N}_M = \langle P_M, I_M \rangle$ and $\mathcal{N}_S = \langle P_S, I_S \rangle$, the procedure *Connected-subgraphs Extraction*, illustrated in Section 3, is

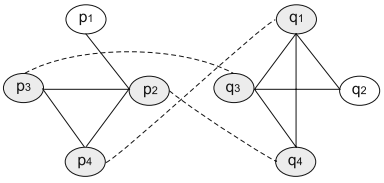


Fig. 6. The matching between the two paired connected subgraphs

called iteratively on the two input networks, at each iteration discarding from the analysis protein nodes belonging to the current solution. The process stops when no further correspondence between pairs of subgraphs is returned. Discarding nodes means eliminating them and all the associated edges from the input networks. This way, a one-to-one correspondence between pairs of nodes in the two networks is constructed.

Figure 7 illustrates a snapshot of the algorithm *Global Alignment*. In detail, the two networks $\mathcal{N}_M$ and $\mathcal{N}_S$ and an integer k are provided in input, and the output solution $\mathcal{S}$ is set equal to the empty-set at the beginning. Then, the procedure *Connected-subgraphs Extraction* is called on $\mathcal{N}_M$, $\mathcal{N}_S$ and k, and the solution S_i it returns is added to $\mathcal{S}$. At this point, nodes included in S_i and all the associated edges are eliminated from the two networks, and *Connected-subgraphs Extraction* is called again until it does not return any further solution. The final $\mathcal{S}$ returned in output will consist in a set of correspondences between pairs of (non-overlapping) connected subgraphs of $\mathcal{N}_M$ and $\mathcal{N}_S$.

5 Experimental Results

We tested our technique on the two PPI networks of *Saccharomyces cerevisiae* (yeast) and *Drosophila melanogaster* (fly). We exploited interaction data collected from BIOGRID [2] and DIP [14]. In particular, the resulting yeast network has 5,443 nodes and 31,898 interactions, while the fly network has 7,404 nodes and 25,830 interactions. The size of the two interaction datasets highlights that the yeast is better characterized than the fly, since a smaller number of fly interactions has been discovered although *D. melanogaster* has a larger number of proteins than *S. cerevisiae*. This is also confirmed by the larger amount of documentation available for the yeast.

We run BLAST [1] to compute the basic similarity dictionary D containing the sequence similarity of pairs of proteins in the two networks. In particular, we exploited the BLAST bit-score to measure protein sequence similarity.

We performed two different series of experiments, in both cases comparing our results with those returned by one of the most successful tools for global alignment, that is, *IsoRank* [12]. *IsoRank* is based on the eigenvalue concept similar to that of the Google PageRanking. It works in two stages: first associate a score with each possible match between nodes of the two networks, and then construct the mapping for the global network alignment by extracting mutually-consistent matches according to a bipartite graph weighted matching performed on the two entire networks.

In the first series of tests, we set the yeast network as the Master and the fly network as the Slave. Then, analyzed things the other way around. In both cases we fixed $k = 2$,

Global Alignment
Input:
- a basic protein similarity dictionary D
- two PPI networks $\mathcal{N}_M = \langle P_M, I_M \rangle$ and $\mathcal{N}_S = \langle P_S, I_S \rangle$
- an integer k

Output: a set $\mathcal{S} = \{S_1, S_2, \ldots, S_p\}$, where each S_i is a set of node pairs representing the correspondence between two connected subgraphs of $\mathcal{N}_M$ and $\mathcal{N}_S$

1: **set** $\mathcal{S} = \emptyset$
2: **repeat**
3: **call** *Connected-subgraphs Extraction* on $\mathcal{N}_M$, $\mathcal{N}_S$, k and D obtaining $S_i = \{(p'_1, q'_1), \ldots, (p'_m, q'_m)\}$
4: **set** $\mathcal{S} = \mathcal{S} \cup \{S_i\}$
5: **set** $P_M = P_M - P'_M$, where $P'_M = \{p'_1, p'_2, \ldots, p'_m\}$
6: **set** $I_M = I_M - I'_M$, where I'_M is the set of edges associated with nodes in P'_M
7: **set** $P_S = P_S - P'_S$, where $P'_S = \{q'_1, q'_2, \ldots, q'_m\}$
8: **set** $I_S = I_S - I'_S$, where I'_S is the set of edges associated with nodes in P'_S
until $S_i \neq \emptyset$

Fig. 7. Global Alignment

and exploited a threshold value of 40.00 on the sequence similarity in order to discard those pairings corresponding to low biological meaning.

5.1 The Yeast as the Master and the Fly as the Slave

When the yeast PPI network has been set as the Master, our system returned a global alignment involving 945 protein pairings, with BLAST similarity bit-scores in the range [45.0, 820.5]. This confirms that the two organisms are not too much related from the evolutionary point of view.

On the same PPI networks, *IsoRank* returned a global alignment involving 5, 499 protein pairings. The fact that the alignment returned by our approach involves a smaller set of pairings is due to the threshold value that we forced on the sequence similarity. In fact, relaxing that constraint the number of returned protein pairs becomes larger. Although when, as in the discussed case, the global alignment returned by our system involves a smaller set of pairings than *IsoRank*, our system returned pairings (in this case, 764 pairings) that *IsoRank* did not. On the other hand, all those pairings returned by *IsoRank* but not by our system have sequence similarity lower then the threshold value. Theese results point out that aligning the two networks from a different point of view, where the approximation plays different roles on the two sides and only what of the Master is conserved in the Slave is searched for, leads to different and still biologically meaningful results.

Table 1 illustrates the top 20 pairings, if results are ordered by protein sequence similarity. In particular, both SWISSPROT ids and protein names, when they were available,

Table 1. The top 20 pairings for yeast as Master and fly as Slave

Yeast ID	Yeast name	Fly ID	Fly name	Similarity
Q00711	*sdh1*	*Q94523*	*scs*	*820.50*
P39533	aconitase	Q9VIE8	aconitase	798.50
P22202	ssa4	O97125	hsp68	798.00
P39007	stt3	Q9XZ53	stt3	741.50
P36022	dynein hc	Q9U3Y5	dynein hc	717.00
P41810	*sec26*	*P45437*	*coatomer subunits beta*	*692.50*
P00830	*atp2*	*Q05825*	*atp2*	*690.00*
P23337	gsy1	Q9VFC8	gsy	639.50
P15274		Q76NQ9		617.50
P53319		P41572	pgd	610.50
P38697		Q07152		599.00
P38972	ade6	P35421	ade2	597.00
Q05931	ssq1	P29845	hsp70	573.00
P32770	arp1	Q9W1G0	tal	570.00
P16862	pfk2	P52034	pkf	570.00
P19882	hsp60	Q9VMN5	hsp60	556.00
P00890		Q9W401	kdn	537.00
P32563	vph1	Q9XZ10	vha	534.50
Q08822	yor356w	Q7JWF1		531.50
P32863	rad54	O76460	okra	528.00

are shown, together with the sequence similarity between the pair of associated proteins. The only three of these pairings found also by *IsoRank* are outlined in Italic.

The yeast proteins shown in Table 1 are correctly paired by our system with the fly homolog when available or with a very similar counterpart. The proteins aligned include enzymes involved in carbohydrate metabolism (*P*16862 / *P*52034; *P*00890 / *Q*9*W*401; *P*23337 / *Q*9*VFC*8), mitochondrial enzymes involved in various metabolic pathways (*Q*00711 / *Q*94523; *P*00890 / *Q*9*W*401; *Q*08822 / *Q*7*JWF*1), glycosyl trasferase (*P*39007 / *Q*9*XZ*53) and other enzymes, but also chaperonin proteins (*P*22202 / *O*97125; *Q*05931 / *P*29845; *P*19882 / *Q*9*VMN*5) and proteins involved in endocytosis (*P*36022 / *Q*9*U*3*Y*5; *P*41810 / *P*45437) are part of the graph.

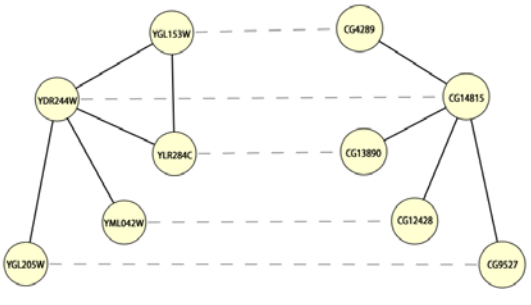


Fig. 8. One of the associated pair of connected subgraphs

Table 2. The top 20 pairings for fly as Master and yeast as Slave

Fly ID	Fly name	Yeast Id	Yeast name	Similarity
P41572	**pgd**	**P53319**		**610.50**
Q76NQ9		***P15274***		***617.50***
P91660	cg8799	P39109	ycf1	628.50
P52029	cg8251	P12709		628.50
Q9VFC8	***gsy***	***P23337***	***gsy1***	***639.50***
A8Y5B7	*kl3*	*P36022*	*dynein*	*643.50*
Q9VUF8	shd	P10964	rpa1	660.00
Q05825	***cg11154***	***P00830***		***690.00***
Q9VKJ6	*atpase*	*P39986*	*atpase1*	*707.50*
Q9XZ53	**stt3**	**P39007**	**stt3**	**741.50**
A8JNX2	*spock*	*P13586*	*pmr1*	*748.00*
P29844	hsp3	P16474	kar2	757.50
Q8IP94	aats-thr	P04801	aats-thr	786.00
Q9VIE8	**cg9244**	**P39533**		**798.50**
Q8IQQ0	*cg11661*	*P20967*		*846.50*
O96553	c1-thf	P07245		920.00
Q0E993	*aats-val*	*P07806*	*aats-val*	*984.00*
P48591	rnrl	rnr3		1029.00
P25167	*rpiii128*	*P22276*	*rpc2*	*1253.00*
Q9VVA4	cg9674	Q12680		1940.00

Figure 8 displays one of the pairs of connected subgraphs associated during the alignment process. In particular, some of the considered proteins are probably involved in protein import into peroxisome matrix and fatty acid beta-oxidation.

5.2 The Fly as the Master and the Yeast as the Slave

When the fly has been exploited as the Master and the yeast as the Slave, the system returned a global alignment involving 707 pairings. Also in this case, there are 589 pairings that *IsoRank* did not returned.

This series of experiments allowed us to make some interesting considerations. In fact, when the focus is turned on the fly network, and most of its structural information are kept, the resulting alignment is smaller than in the previous case. This is possibly due in part to the fact that the yeast is better characterized than the fly, thus, it presents a larger number of interactions. When the yeast is the Slave, most of its structural information gets lost, and thus some of the associations found in the previous case are no longer recognized.

A second key to explain the results is the following. When a PPI network is exploited as the Master, this makes the search process to follow a precise direction, that is, searching for *those regions of the Master which have been conserved in the Slave*. Our analysis shows that, according to the available interaction data, there are more yeast regions that have been conserved in the fly than vice versa, which is reasonable observing that the fly is a more complex organism than the yeast.

Another aspect to consider is that, also in this verse, the system is able to return significative associations. Table 2 illustrates the top 20 pairings w.r.t. protein sequence similarity, where only nine pairings (highlighted in Italic) have been returned also by *IsoRank*. Note that six pairings, pointed out in bold, have been found also in the case yeast-Master and fly-Slave. Looking at Table 2 it is possible to observe that our system correctly pairs most conserved proteins that include, as in the previous example, mainly metabolic enzymes (*P*41572 / *P*53319; *Q*76*NQ*9 /*P*15274; *P*52029 / *P*12709; *CG*9244 / *P*39533; *Q*8*IQQ*0 / *P*20967), glycosyl trasferase (*P*39007 / *Q*9*XZ*53), aminoacil-tRNA- synthetase (*Q*8*IP*94 / *P*04801; *Q*0*E*993 / *P*07806) that are crucial enzymes for protein synthesis, and RNA polymerase subunits (*P*25167 / *P*22276). Yeast dynein heavy chain, that in Table 1 was paired with the fly homolog, here is paired with the fly male fertility factor kl3. Nevertheless, this unknown fly factor is probably a dynein subunit because of its molecular features and its inferred GO annotations.

The two global alignments obtained for the two different settings of the Master and the Slave share 181 pairings.

6 Conclusions

We proposed a method for global alignment of PPI networks, based on a "Master-Slave" approach. In particular, one of the two input networks is set as the "Master", and the other one as the "Slave". The difference between Master and Slave is that most of the Master structural information, suitably encoded by a finite state automaton, are kept and exploited during the alignment process, while the Slave is linearized in order to be considered as a possible output of the automaton. The goal of the approach is that of using the Master as a guide for the global alignment to be performed, in order to search for those regions of the Master that are conserved in the Slave. Experimental results on the two networks of *Saccharomyces cerevisiae* and *Drosophila melanogaster* showed that our technique is able to find significant pairings, and confirmed that exchanging the Master with the Slave the alignment process takes different directions.

The method is general enough to be applied to other types of networks. Furthermore, the approach can be extended to handle multiple network alignment by iteratively aligning pairs of networks and taking, at any iteration, the set of already aligned networks, encoded as a suitable finite state automaton, as the Master. We argue that, when more reliable and accurate interaction data will be available, our approach can effectively support the discovery and prediction of unknown protein functions for the less characterized organisms, providing a new direction of investigation that is orthogonal to those of the other techniques.

References

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

[2] Stark, C., Breitkreutz, B.J., Reguly, T., Boucher, L., Breitkreutz, A., Tyers., M.: Biogrid: a general repository for interaction datasets. Nucleic Acid Research 34(Database Issue), D535–D539 (2006)

[3] Flannick, J., Novak, A., Do, C.B., Srinivasan, B.S., Batzoglou, S.: Automatic parameter learning for multiple network alignment. In: Vingron, M., Wong, L. (eds.) RECOMB 2008. LNCS (LNBI), vol. 4955, pp. 214–231. Springer, Heidelberg (2008)
[4] Forney, G.D.: The Viterbi algorithm. Proceedings of the IEEE 61(3), 268–278 (1973)
[5] Garey, M., Johnson, D.: Computers and intractability: A guide to the theory of NP-completeness. Freeman, New York (1979)
[6] Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M., Sakaki, Y.: A comprehensive two-hybrid analysis to explore the yeast protein interactome. Proceedings of the National Academy of Sciences of the USA 98(8), 4569–4574 (2001)
[7] Kalaev, M., Bafna, V., Sharan, R.: Fast and accurate alignment of multiple protein networks. In: Vingron, M., Wong, L. (eds.) RECOMB 2008. LNCS (LNBI), vol. 4955, pp. 246–256. Springer, Heidelberg (2008)
[8] Kempe, A.: Viterbi algorithm generalized for n-tape best-path search. CoRR, abs/cs/0612041 (2006)
[9] Kiemer, L., Costa, S., Ueffing, M., Cesareni, G.: WI-PHI: A weighted yeast interactome enriched for direct physical interactions. Proteomics 7, 932–943 (2007)
[10] Klau, G.W.: A new graph-based method for pairwise global network alignment. BMC Bioinformatics 10 (suppl. 1), S59 (2009)
[11] Krogan, N.J., Cagney, G., et al.: Global landscape of protein complexes in the yeast saccharomyces cerevisiae. Nature 440(7084), 637–643 (2006)
[12] Liao, C.-S., et al.: Isorankn: spectral methods for global alignment of multiple protein networks. Bioinformatics 25, i253–i258 (2009)
[13] Narayanan, M., Karp, R.M.: Comparing protein interaction networks via a graph match-and-split algorithm. Journal of Computational Biology 14(7), 892–907 (2007)
[14] Salwinski, L., Miller, C.S., Smith, et al.: The database of interacting proteins: 2004 update. Nucleic Acids Reserch 32(Database issue), D449–D451 (2004)
[15] Singh, R., Xu, J., Berger, B.: Pairwise global alignment of protein interaction networks by matching neighborhood topology. In: Speed, T., Huang, H. (eds.) RECOMB 2007. LNCS (LNBI), vol. 4453, pp. 16–31. Springer, Heidelberg (2007)
[16] Singh, R., Xu, J., Berger, B.: Isorank: Global alignment of multiple protein interaction networks with applications to functional orthology detection. Proceedings of the National Academy of Sciences 105(35), 12763–12768 (2008)
[17] Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Trans. Inform. Theory IT–13, 260–269 (1967)
[18] von Mering, D., Krause, C., et al.: Comparative assessment of a large-scale data sets of protein-protein interactions. Nature 417(6887), 399–403 (2002)

Deciphering Transcription Factor Binding Patterns from Genome-Wide High Density ChIP-chip Tiling Array Data

Juntao Li[1], Lei Zhu[2], Majid Eshaghi[2],
Jianhua Liu[2], and Radha Krishna Murthy Karuturi[1,⋆]

[1] Computational & Mathematical Biology, Genome Institute of Singapore, Singapore
[2] Systems Biology, Genome Institute of Singapore, Singapore
{lij9,zhul,eshaghim,liujh,karuturikm}@gis.a-star.edu.sg

Abstract. The binding events of DNA-binding proteins can be extensively characterized by high density ChIP-chip tiling array data. The binding sites and binding occupancy patterns are all very useful to understand the DNA-protein interaction. We propose a statistical procedure which focuses on identifying the interaction signal regions and the patterns of interaction using peakedness and skewness tests. Its utility to annotate the binding signals by analyzing the Tbp1 and Rpb1 ChIP-chip datasets in fission yeast is demonstrated.

Keywords: ChIP-chip, Tiling Array, Transcription factor, Kurtosis, Skewness.

1 Introduction

With the chip technology rapidly advanced, tiling arrays have quickly become one of the most powerful tools in genome-wide investigations. High density tiling arrays [1] can be used to address many biological problems such as transcriptome mapping, protein-DNA interaction mapping (ChIP-chip) and array CGH among others [2]. ChIP-chip [3],the focus of the paper, is a technique that combines chromatin immunoprecipitation (ChIP) with microarray technology (chip). It allows the identification of binding sites of DNA-binding proteins in a very efficient and scalable way [4]. High density ChIP-chip tiling arrays not only help us map the binding locations of a protein in the genome, but also allow us to fully understand the binding events of the protein by clearly displaying the binding occupancy patterns.

Several methods have been proposed to analyze the ChIP-chip data; for example, Joint Binding De-convolution (JBD) [5] uses a probabilistic graphical model to improve spatial resolution of identification of the transcription factor binding sites. However it requires the DNA fragment length distribution which may not always be available. It may not be so useful for high density tiling array

⋆ Corresponding author.

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 230–240, 2010.

since the resolution is already considerably high. MPeak [6,7] fits a mixture of triangular basis to model the binding or interaction data. It ignores the complexity of binding event and only roughly characterized the basic patterns for single and direct binding events. The more complex binding patterns from high density ChIP-chip may not be well modeled using mixture of a triangular basis.

We propose a new statistical procedure to analyze high density ChIP-chip tiling array data to characterize protein-DNA interaction. First, we identify the enriched signal regions or protein binding occupancies using moving window binomial analysis and split the signal regions with multiple peaks into individual peak regions. The signal regions are classified into two categories using peakedness test and process them separately. The peak regions are processed to get the peak positions signifying binding locations, and using skewness test to improve the peak assignment to genes. The flat binding occupancies are processed to summarize their overall strength.

In this article, we applied our procedure to analyze the data of fission yeast (*Schizosaccharomyces pombe*) from custom designed NimbleGen genome tiling arrays of $\sim 380k$ probes. We studied one general transcription factor Tbp1 (TATA box binding protein) together with the RNA polymerase II large subunit Rpb1, which is used to indicate transcriptionally active genes, of *S.pombe*. We found that DNA-binding proteins show distinct patterns in the proportion of sharp and flat bindings. Tbp1 shows more sharp binding patterns indicating its location specific binding, and Rpb1 presents a large fraction of flat signal regions indicating variability of its binding.

2 Method

2.1 Data Preprocessing

The tiling array has very high resolution and probes cover the whole genome, and ChIP procedure selects only protein binding sites which are a small part of the genome. Therefore, only a very small proportion of probes in tiling array has the binding signal and majority probes'signals will be close to the background. Hence we median centered the log transformed data for further analysis.

Two different smoothing methods, multiple round moving average and median smoothing, are employed to reduce the noise in data preprocessing. Multiple round moving average smoothing method will retain the signal shape and the peak signal (local maximum) loci. This method has already been used in the ChIP-chip peak finder [8]. We use moving average method to identify the peak loci and compute kurtosis and skewness of signal regions. The drawback of moving average is that it may destroy the boundaries of the signal regions, so we apply moving median smoothing method to characterize signal regions.

2.2 Moving Window Binomial Analysis

For any location i, let x_i $(i = 1, \ldots, n)$ denote the median centered log data and we first define the base threshold as cMAD, c fold MAD (median absolute deviation), of x_i $(i = 1, \ldots, n)$, and

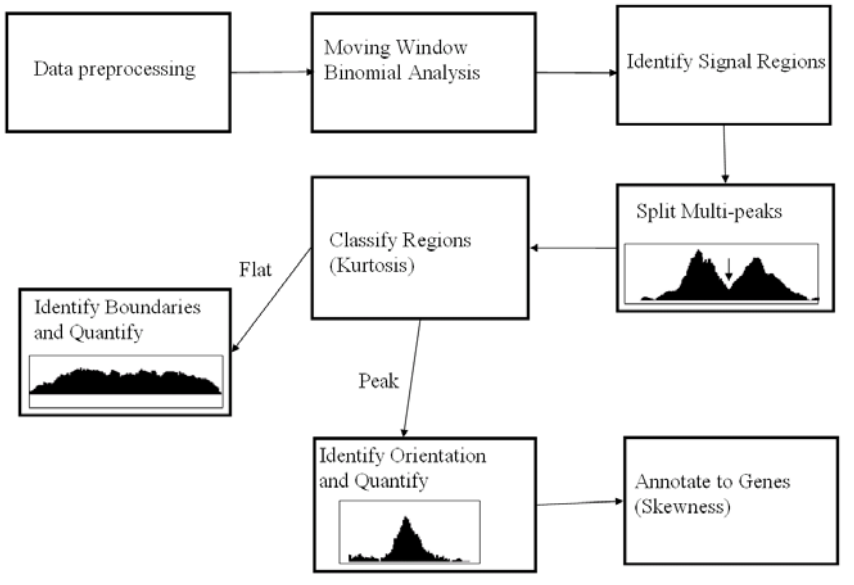


Fig. 1. The proposed statistical procedure for ChIP-chip data analysis

$$p = \frac{\#\{x_i | x_i \geq c\text{MAD}\}}{n}$$

is the probability that a single probe pass the base threshold. Then the region $\{x_i\}_{i-w}^{i+w}$ is the binomial sequence with signal probability p for each x_i, where w is the predefined half window size. $p_w(x_i)$ is defined as the probability that x_i is classified as signal by considering the region $\{x_i\}_{i-w}^{i+w}$ as the signal region, and can be computed in the following equation,

$$p_w(x_i) = \sum_{i=C}^{2w+1} \binom{2w+1}{i} p^i (1-p)^{2w+1-i}$$

where C is the number of probes above the base threshold in region $\{x_i\}_{i-w}^{i+w}$.

We define a region $\{x_i\}_s^e$ as a signal region if

$$\begin{cases} p_w(x_i) < \alpha, (s \leq i \leq e) \\ e - s \geq 4 \\ x_s \geq c\text{MAD}, \, x_e \geq c\text{MAD} \end{cases}$$

where α is the p-value cutoff of the binomial test.

2.3 Region Splitting for Multi-peak Region

Each signal region from moving window binomial analysis may contain multiple peaks (local maxima), and this will make the binding regions pattern more

complex. Therefore, the regions with multiple peaks are split at the troughs of the signal region's profile. For doing this, we first assign each probe one of the $\gamma_{x_i} = \{+, -, 0\}$ to indicate whether the binding signal are significantly increased, decreased or not significantly changed,

$$\gamma_{x_i} = \begin{cases} + & \text{if } x_{i+1} - x_i > +d \\ - & \text{if } x_{i+1} - x_i < -d \\ 0 & \text{otherwise.} \end{cases}$$

where $d = \text{MAD}(\delta_i)$, $\delta_i = |x_i - x_{i+1}|$ for $i = 1, 2, \ldots, n-1$. After removing all probes with "0", the region is split between the opposite signs such as from "-" to "+". After splitting, the signal regions which have less than 4 probes are removed.

Peak position of a signal region with one significant peak is determined after the moving average smoothing of the profile. The position of probe with the maximal value in this region is defined as the position of the peak.

2.4 Peakedness and Skewness Test for Signal Region

The signal regions $\{x_i\}_s^e$ are tested for peakedness using kurtosis based on the formula

$$K = \frac{\sum_{j=s}^{e}(j-u)^4 p_j}{[\sum_{j=s}^{e}(j-u)^2 p_j]^2}$$

where $u = \sum_{j=s}^{e} j p_j$ and $p_j = \frac{x_j}{\sum_{i=s}^{e} x_i}$ $(j = s, s+1, \ldots, e)$.The region is designated as having flat-shaped when its $K < 2$. Thus, peaked regions are separated from flat regions.

For single peak regions, we used skewness score to test whether the peaks are left-skewed or right-skewed which indicates the binding orientation. This is very useful for transcription factor binding assignment to a single gene if there is a binding region in a bidirectional intergenic region (intergenic regions from divergent pair of genes). The skewness score is based on the formula

$$G = \frac{\sum_{j=s}^{e}(j-u)^3 p_j}{[\sum_{j=s}^{e}(j-u)^2 p_j]^{3/2}}$$

where the u and p_j have same definition as kurtosis.

3 Results

3.1 Probe Design for Tiling Array in Fission Yeast

We used customized NimbleGen Tiling array which has $\sim 380k$ $50mer$ probes. They cover both strands of entire *S. pombe* genome based on the genome sequence from Wellcome Trust Sanger Institute (ftp://ftp.sanger.ac.uk/pub/yeast/pombe/). In each strand, there is a $16bp$ interval between two consecutive probes. The probes on the reverse strand are placed so that they cover

the gaps between consecutive probes of the pairing forward strand. Therefore, the probes have 17*bp* overlap with each other without considering the strand-specificity. The probes with more than 4 hits in the genome were removed in the analysis, $\sim 2\%$ of probes have been removed.

3.2 Transcription Factor Binding Regions and Binding Patterns

We analyzed ChIP-chip experiments of general transcription factors Tbp1 together with the RNA polymerase II large subunit Rpb1 which indicates transcriptionally active genes. Tbp1 is a core subunit of the eukaryotic transcription factor TFIID, binding specifically to the TATA box. It contributes to load and release of RNA polymerase II at the transcription start sites (TSS). Furthermore, Tbp1 is also a necessary component of RNA polymerase I and RNA polymerase III. Therefore, Tbp1 is a good choice for binding pattern study. There are two replicates for each ChIP-chip experiment.

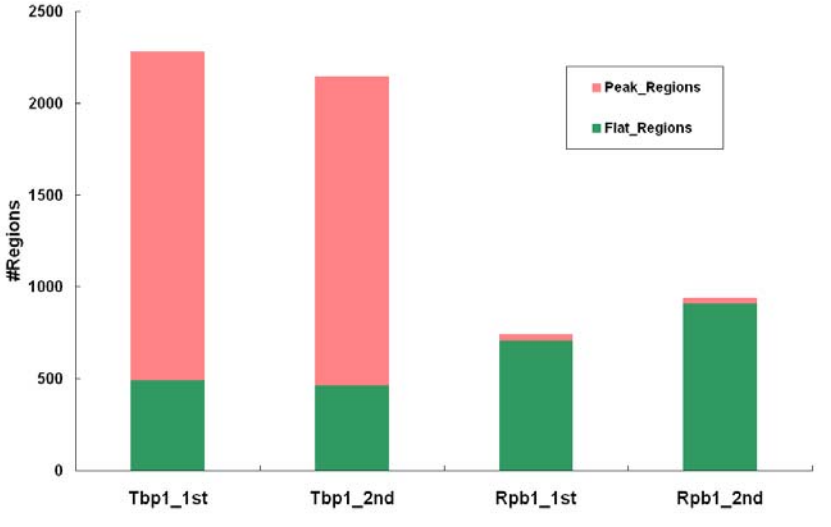


Fig. 2. The signal regions summary for 4 ChIP-chip datasets

The signal regions for each array were identified with the stringent criteria of cMAD= 2MAD at p-value less than $0.001(\alpha = 0.001)$. The number of signal regions is $\sim 2000(\sim 1500$ before split) for each replicate of Tbp1, and for Rpb1 it is $\sim 800(\sim 500$ before split). We applied Dice coefficient to measure the similarity of signal regions between first and second repeats.

$$S = \frac{2|A \cap B|}{|A| + |B|}$$

where $|A|$ and $|B|$ is the total length of all signal regions of the first and second repeats, $|A \cap B|$ is the length of their overlapping regions. The coefficient for

Tbp1 is 0.921 which indicates that our result of Tbp1 signal regions are highly reproducible. The coefficient for Rpb1 is 0.743 which is still considerably high.

The summary of the peakedness test is shown in Figure 2. Majority of the Tbp1 signal regions are sharp peaks and Rpb1 signal regions are mostly flat. It is consistent with our knowledge about the protein characters of Tbp1 and Rbp1. Since the purpose of performing a ChIP-chip experiment is to transform transcription factor binding sites into IP-enriched DNA, the specificity of protein-DNA binding finally determines the peaks of IP-enrichment. Therefore, due to the specific binding affinity to TSS position, Tbp1 were observed binding to DNA with many sharp peaks. However, Rpb1 mostly presents flat occupancies in coding regions that is because of the function of Rpb1 which controls transcription elongation and synthesize messenger RNAs.

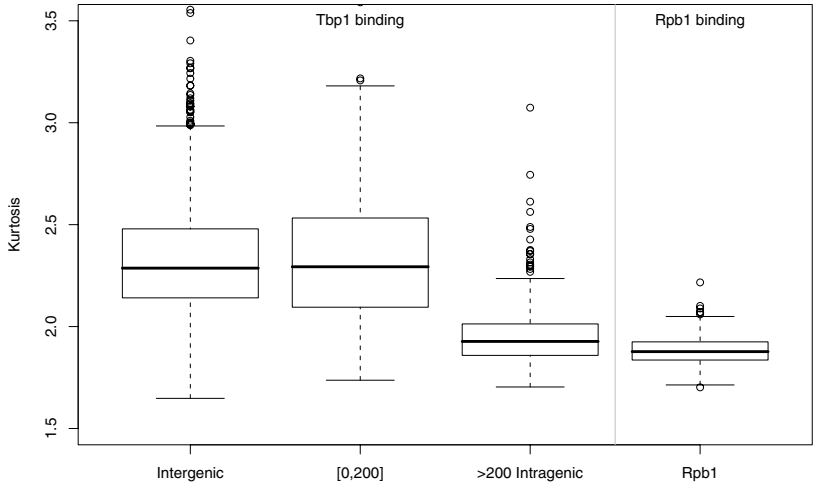


Fig. 3. The kurtosis score for Tbp1 binding in intergenic, [0,200] intragenic and $> 200bp$ intragenic regions, and kurtosis score for Rpb1 binding

In order to investigate the kurtosis score distribution in different genomic regions, we examined kurtosis score in intergenic and intragenic regions. Due to tiling array probe and DNA fragment length, we separated intragenic regions into two groups, the coding regions less than and more than $200bp$ from translation start site. As shown in Figure 3, the signal regions in [0,200] of coding regions and intergenic regions have similar high kurtosis score. This indicate that the Tbp1 biding to specific biding sites in these two regions and involve transcription initiation. However, the Tbp1 signal regions falling into coding regions away by more than $200bp$ from start site mostly have low kurtosis. The low kurtosis regions in coding regions are probably involve transcription elongation, and the peak loci in these regions are general weak and unstable, may not refer the exact binding sites. The Rpb1 binding regions mostly fall into coding regions and present the flat patterns.

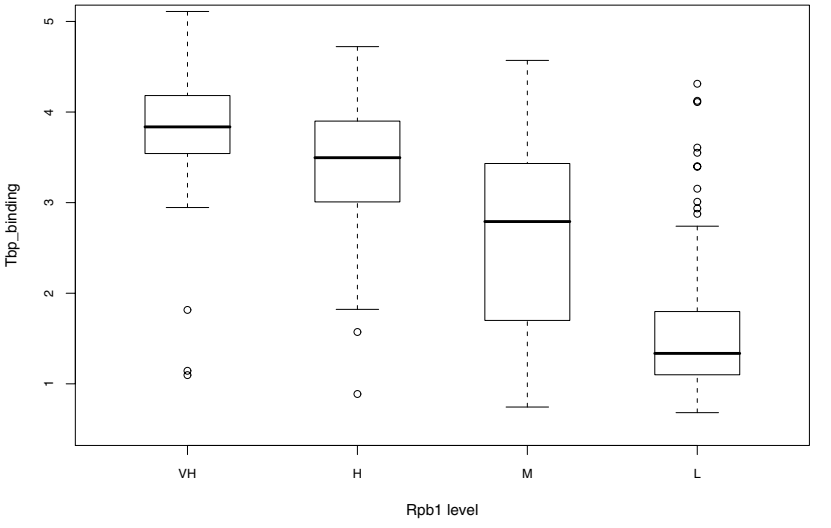


Fig. 4. Tbp1 binding affinity for VH(very high), H(high), M(Median), L(Low) Rpb1 level groups

3.3 Tbp1 Binding Affinity Positively Correlates with the Gene Transcription Level

After having identified all signal regions from ChIP-chip data, the next step is mapping the regions to the genes. To our knowledge, there is no perfect method to accurately map binding sites to the genes. To do so, we have limited our investigation to the peaks only from the unidirectional intergenic (IGU) regions which are easy to assign, i.e. assigned to the downstream genes, to reduce the risk of assignment errors. Furthermore, we filtered out peaks not within the upstream $1kb$ of any gene as it may be out of the promoter regions for S.pombe. The upstream peaks of any RNA genes have also been removed since Tbp1 is also associated with RNA polymerase I and RNA polymerase III. There are 379 peaks in unidirectional intergenic regions (IGU peaks) left after filtering. Then we investigated the Tbp1 binding affinity for those peaks. The Rpb1 level of each Tbp1 binding gene is the median level of Rpb1 occupancies within the ORF, which measures the level of the transcriptional activity. From Figure 4, we observed positive correlation between Tbp1 binding affinity and transcription levels of the protein-coding genes. It implies that the highly transcribed genes tend to be initiated by high Tbp1 binding affinity at their promoters.

3.4 Skewness of Tbp1 Binding Regions Helps Identifying Tbp1 Regulated Genes

The skewness scores of Tbp1 binding regions also positively correlate with Rpb1 occupancy levels. We investigated the skewness for 379 peaks in unidirectional

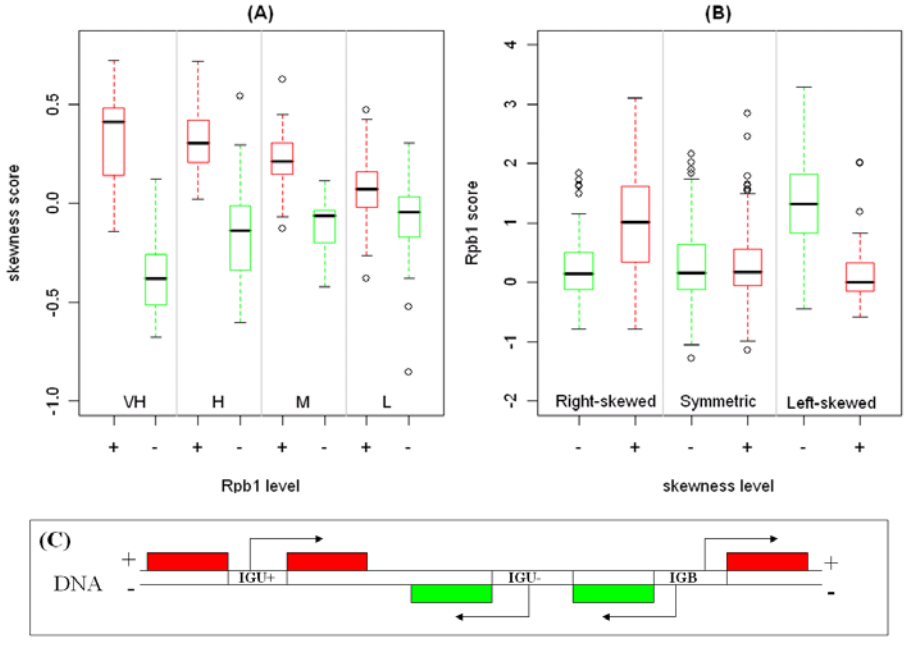


Fig. 5. (A)The skewness score of Tbp1 binding in IGU (unidirectional intergenic) regions for VH(very high), H(high), M(Median), L(Low) Rpb1 level groups. Red boxes are the IGU+ regions and green boxes are IGU- regions. (B)The Rpb1 score for Right-skewed,Symmetric, Left-skewed groups in IGB (bidirectional intergenic) regions. Red boxes are genes in forward strand and green boxes are genes in reverse strand. (C)The illustration of the IGU in forward strand (IGU+) and reverse strand (IGU-) and IGB regions. Red boxes are genes in forward strand and green boxes are genes in reverse strand.

intergenic regions (IGU peaks). Interestingly, the patterns of Tbp1 binding skewed towards the direction of immediate downstream gene transcribed. in another words, Tbp1 signal region extend a tail into the ORF of its target gene. As shown in Figure 5(A), transcribed genes on the forward strand tend to display positive skewness with Tbp1 binding regions in their promoters, and the transcribed genes on the reverse strand preferentially show negative skewness. Most interesting observation is that the absolute skewness declines with the decreasing Rpb1 level of the downstream genes indicating that Tbp1 binding pattern is enough to predict whether the downstream genes are transcribed in that condition. Our explanation is that Tbp1 can be pulled into transcribing regions by RNA polymerase complex during the transition between transcription initiation and elongation, and the amount of pull on Tbp1 would correlate with the transcription rate of the downstream genes.

To further test our observations and demonstrate the utility of the skewness of binding regions, we checked the correlation between skewness of Tbp1 binding in the bi-directional promoters and Rpb1 levels of the flanking genes. The assignment

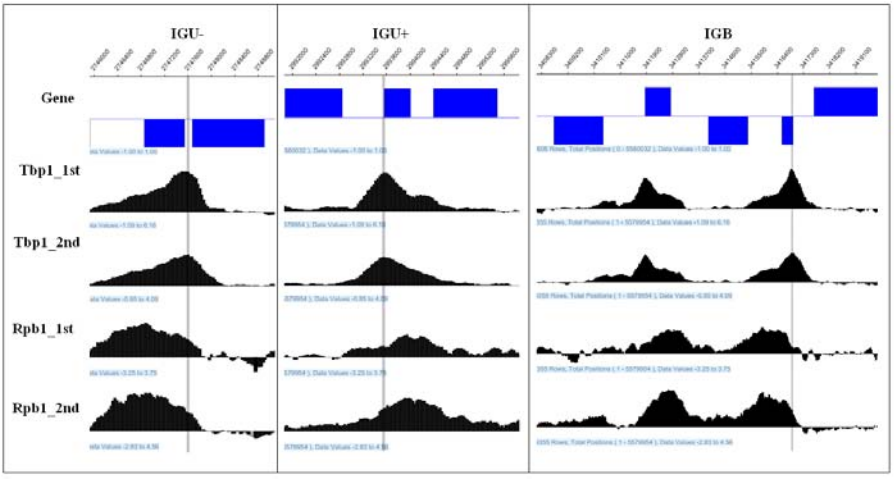


Fig. 6. Examples of Tbp1 binding patterns and Rpb1 occupancies with two repeats in IGU+, IGU- and IGB regions

of binding sites in the bidirectional promoters is always a problem: some studies assign them to both genes while the others assign them to the nearest gene. Here, we found skewness score may help us to get a better assignment i.e. to identify the gene activated by Tbp1. In order to be conservative, we removed bidirectional intergenic regions with one of the flanking genes is an RNA gene and also discarded those gene pairs having more than one tbp1 binding peaks. Finally there are 387 gene pairs used in the analysis. As shown in Figure 5(B), when the skewness scores are significantly positive (greater than 0.15) then the transcription level of the genes on the forward strands is clearly higher compared to the corresponding paired gene on the reverse strand, vice versa.

When the binding pattern seems symmetric (between -0.15 and 0.15), there are no significant difference for transcription level between genes on the forward strand and the reverse strand. In addition, as the Figure 5 shown, the presence of symmetric pattern is associated with the Rpb1 level less than 0.5 i.e. there is almost zero transcription events for such low level transcriptions are rarely detectable with our Rpb1 data under 2MAD cutoff. Therefore, the skewness of peaks could be helpful in assigning Tbp1 binding to annotate features, particularly for the binding sites located in IGB (bidirectional intergenic) regions. Some examples of Tbp1 binding patterns and Rpb1 occupancies with two repeats in IGU+, IGU- and IGB regions are shown in Figure 6.

4 Discussion

We developed a statistical procedure to characterize binding events of DNA-interacting proteins especially transcription factors from high density ChIP-chip

tiling array data. The signal regions are detected using moving window binomial analysis and the binding events are characterized by two shape parameters, Kurtosis and Skewness.

We applied our method to ChIP-chip experiments of TATA box binding protein (Tbp1) and Rpb1 in S.pombe. We found that Tbp1 tend to have more sharp peaks than Rpb1 that indicates our methods can efficiently distinguish mostly localized DNA-protein bindings and the scattered DNA-protein interactions. We should notice Tbp1 also has flat bindings, that maybe due to the interaction of Tbp1 with RNA polymerase complex are maintained after the event of transcription initiation since other studies have reported there is no stall of RNA polymerase II at the promoter regions in yeast [9]. It is also possible that Tbp1 or other components of TFIID behavior functionally during transcription elongation. It needs more lab experiments to discover.

The two shape parameters of the signal regions, kurtosis and skewness, can characterize the binding patterns. We used kurtosis to classify the regions into peak and flat regions. The peak regions mostly fall into the promoters and the flat regions are mostly very large and cover the coding regions. We have demonstrated that the binding pattern of the peak regions in promoter regions are skewed to the downstream genes if they are transcribed, and hence the skewness can help us to predict whether the downstream gene is transcribed and to assign the binding sites to genes in the bidirectional intergenic regions.

Our method is applicable not only to ChIP-chip data, but also to other datasets with similar goal. For example, ChIP-seq and ChIP-chip measure same signals but with different techniques. The binding patterns for ChIP-seq data should be similar to ChIP-chip data, so the peakedness and skewness tests can be used for further analysis. Our method can be extended to other tiling array data if the patterns of the signal regions are important to the corresponding studies.

Acknowledgments

The authors thank Edison T. Liu for his constant encouragement and support during this work. We appreciate Jonghoon Lee, Puteri Paramita, Rohini Parameswarath for their valuable discussion. This work was supported by the Genome Institute of Singapore, Biomedical Research Council, Agency for Science, Technology and Research (A*STAR), Singapore.

References

1. Mockler, T.C., Chan, S., Sundaresan, A., Chen, H., Jacobsen, S.E., Ecker, J.R.: Applications of DNA tiling arrays for whole-genome analysis. Genomics 85(1), 1–15 (2005)
2. Yazaki, J., Gregory, B.D., Ecker, J.R.: Mapping the genome landscape using tiling array technology. Current Opinion in Plant Biology 10(5), 534–542 (2007)
3. Buck, M.J., Lieb, J.D.: ChIP-chip: considerations for the design, analysis, and application of genome-wide chromatin immunoprecipitation experiments. Genomics 83(3), 349–360 (2004)

4. Iyer, V.R., Horak, C.E., Scafe, C.S., Botstein, D., Snyder, M., Brown, P.O.: Genomic binding sites of the yeast cell-cycle transcription factors SBF and MBF. Nature 409(6819), 533–538 (2001)
5. Qi, Y., Rolfe, A., MacIsaac, K.D., Gerber, G.K., Pokholok, D., Zeitlinger, J., Danford, T., Dowell, R.D., Fraenkel, E., Jaakkola, T.S., Young, R.A., Gifford, D.K.: High-resolution computational models of genome binding events. Nature Biotechnology 24(8), 963–970 (2006)
6. Kim, T.H., Barrera, L.O., Zheng, M., Qu, C., Singer, M.A., Richmond, T.A., Wu, Y., Green, R.D., Ren, B.: A high-resolution map of active promoters in the human genome. Nature 436(7052), 876–880 (2005)
7. Zheng, M., Barrera, L.O., Ren, B., Wu, Y.N.: ChIP-chip: data, model, and analysis. Biometrics 63(3), 787–796 (2007)
8. Glynn, E.F., Megee, P.C., Yu, H., Mistrot, C., Unal, E., Koshland, D.E., DeRisi, J.L., Gerton, J.L.: Genome-wide mapping of the cohesin complex in the yeast saccharomyces cerevisiae. PLoS Biology 2(9), E259 (2004)
9. Wade, J.T., Struhl, K.: The transition from transcriptional initiation to elongation. Current Opinion in Genetics & Development 18(2), 130–136 (2008)

The Expected Fitness Cost of a Mutation Fixation under the One-Dimensional Fisher Model

Liqing Zhang[1] and Layne T. Watson[2]

[1] Department of Computer Science,
Virginia Polytechnic Institute & State University,
Blacksburg, VA 24061, USA
lqzhang@cs.vt.edu
[2] Departments of Computer Science and Mathematics,
Virginia Polytechnic Institute & State University
Blacksburg, VA 24061, USA
ltw@cs.vt.edu

Abstract. This paper employs Fisher's model of adaptation to understand the expected fitness effect of fixing a mutation in a natural population. Fisher's model in one dimension admits a closed form solution for this expected fitness effect. A combination of different parameters, including the distribution of mutation lengths, population sizes, and the initial state that the population is in, are examined to see how they affect the expected fitness effect of state transitions. The results show that the expected fitness change due to the fixation of a mutation is always positive, regardless of the distributional shapes of mutation lengths, effective population sizes, and the initial state that the population is in. The further away the initial state of a population is from the optimal state, the slower the population returns to the optimal state. Effective population size (except when very small) has little effect on the expected fitness change due to mutation fixation. The always positive expected fitness change suggests that small populations may not necessarily be doomed due to the runaway process of fixation of deleterious mutations.

Keywords: Fisher's model, effective population size, compensatory mutation, generalized Riemann zeta function, incomplete gamma function.

1 Introduction

The statistician R. Fisher [2] proposed a geometrical model to understand the nature of adaptation. The basic idea of his model can be illustrated using a simple one-dimensional system. Imagine that a trait has the optimal state at the origin, the population's current state can be represented by point A on the real coordinate line, and the distance between point A and the origin O represents the fitness of the population at state A. Mutations can occur with both magnitude and direction, which will drive the population either further

M. Borodovsky et al. (Eds.): ISBRA 2010, LNBI 6053, pp. 241–252, 2010.

away from the population optimum point O, or towards the optimum point O. One can therefore model the dynamics of mutations by tracking the movement of the population states owing to the fixation of mutations.

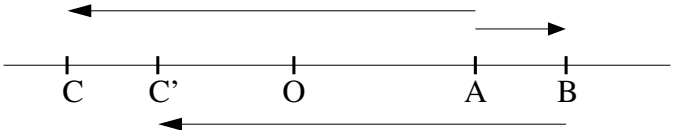


Fig. 1. Fisher's model of adaptation in one dimension

The attractiveness of Fisher's model lies in the fact that it nicely incorporates the nonindependent nature of multiple mutations. For example, in the one-dimensional system, suppose that the population starts at state A, that is, all the individuals in the population carry the allele A. A mutation of a certain type will take the population to state B, where all the individuals in the population carry the mutated type B. Similarly, from state A a mutation of a different type will take the population to state C, where all the individuals carry the mutant type C. Compared with the original state A, both mutations are deleterious and move the population to states (B or C) that have lower fitness than the original state A. However, if both mutants appear and get fixed together, the population will have a fitness gain at state C' from the original state A. Therefore, both mutations are deleterious and reduce the population fitness when fixed individually. However, the joint fixation of the two leads to a fitness gain instead—the two deleterious mutations are compensatory. Therefore, Fisher's model has built-in nonindependence, and elegantly models the nonindependent feature of mutations. Fisher's model of adaptation has been applied to study compensatory mutations by, e.g., Poon and Otto [5], who studied the effect of compensatory mutations with respect to the number of character dimensions. They concluded that the effects of compensatory mutations become more pronounced when the number of character dimensions increases.

This paper examines the expected fitness cost of transition from one population state to another, using Fisher's model in one dimension, where closed form analytic solutions exist. It has been shown that the n-dimensional Fisher model can be reduced to two dimensions (polar coordinates), for which the marginal distributions are one-dimensional [2], and [4]. Thus the one-dimensional results here apply to the marginal distributions for the general case (n dimensions reduced to two), and are of some interest. Assuming a gamma probability distribution for the mutation magnitude, the present work derives analytically the mean fitness cost of a transition, and studies the effect of a variety of parameters, including the population size and different initial states, on the next state transition. The biological implications of the findings are discussed.

2 Mathematical Derivations

This section focuses on deriving the expected fitness effect of mutations moving the population from one state to another. Because the comparison is between the current population state and the next state, the fitness effect is thus the comparison of these two states. Assume that the distance away from the optimum point (origin on real line) corresponds to the fitness w of the state via the equations $w(A) = e^{-|z|}$ and $w(B) = e^{-|z'|}$, where z and z' are signed real numbers, representing the coordinate positions of population states A and B. The selection coefficient of the mutation from A to B is

$$s = \frac{W(B)}{W(A)} - 1 = \frac{e^{-|z'|} - e^{-|z|}}{e^{-|z|}} = e^{|z|-|z'|} - 1. \approx 1 - \left|\frac{z'}{z}\right| \tag{1}$$

for $|z| \approx |z'|$ and $|z| \approx 1$. The first assumption, $|z| \approx |z'|$, corresponds to $|s| \approx 0$, a common assumption in the literature (that $|s|$ is large with vanishingly small probability). The second assumption, $|z| \approx 1$ for the current population state, corresponds to scaling the distance measure z. It turns out that for $s = 1 - \left|\frac{z'}{z}\right|$ closed form expressions can be derived and that is done below.

Due to the uncertainty about the distribution of mutations, assume that mutation magnitude from one state to another (i.e., $|z' - z|$) is gamma distributed, which incorporates a variety of distribution shapes (with different parameters) and thus models a rich collection of mutation scenarios. Specifically, let the probability density function of mutation to z' from z be

$$f(z') = \frac{|z' - z|^{\alpha-1}\beta^{\alpha}e^{-\beta|z'-z|}}{\Gamma(\alpha)}, \tag{2}$$

where α and β are the shape and location parameters in the gamma distribution. The fixation probability $u(s)$ of the mutation state has been given by Crow and Kimura [1] as

$$u(s) = \frac{1 - e^{-2N_e s/N}}{1 - e^{-4N_e s}}, \tag{3}$$

where N_e is the effective population size, and N is the population size. For simplicity, the analysis here takes $N_e = N$.

Assuming that the magnitude of mutations has a gamma distribution, and the fitness effect of a new mutation depends on the current state of the population mutation, then the gamma probability density function times the fixation probability of the mutation times the fitness change s (for diploid populations, $2s$ is used), integrated over all new states z', yields the *expected (relative) fitness effect of a state transition* from z:

$$W(z) = \int_{-\infty}^{\infty} sf(z')u(s)dz' = W_1(z) - W_2(z), \tag{4}$$

where $\alpha > 0$, $\beta > 0$, $N_e > 0$, and

$$W_1(z) = \int_{-\infty}^{\infty} f(z')u(s)dz', \qquad W_2(z) = \int_{-\infty}^{\infty} \left|\frac{z'}{z}\right| f(z')u(s)dz'.$$

These integrals W_1 and W_2 will be expressed in terms of the gamma function $\Gamma(\alpha)=\int_0^\infty x^{\alpha-1}e^{-x}dx$, the incomplete gamma function $\gamma(\alpha,y)=\int_y^\infty x^{\alpha-1}e^{-x}dx$, the generalized incomplete gamma function $\hat{\gamma}(\alpha,x,y)=\gamma(\alpha,x)-\gamma(\alpha,y)$, and the generalized Riemann zeta function $\mathcal{Z}(s,a)=\sum_{k=0}^{\infty}\frac{1}{(k+a)^s}$.

Because of the absolute values, doing the integrals analytically requires considering different cases. First, consider the case when $z>0$. Write $W_1(z)=D_1+D_2+D_3$, where

$$D_1=\int_{-\infty}^{0}\frac{(z-z')^{\alpha-1}\beta^\alpha e^{-\beta(z-z')}}{\Gamma(\alpha)}\frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}}dz', \tag{5}$$

$$D_2=\int_{0}^{z}\frac{(z-z')^{\alpha-1}\beta^\alpha e^{-\beta(z-z')}}{\Gamma(\alpha)}\frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}}dz', \tag{6}$$

$$D_3=\int_{z}^{\infty}\frac{(z'-z)^{\alpha-1}\beta^\alpha e^{-\beta(z'-z)}}{\Gamma(\alpha)}\frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}}dz'. \tag{7}$$

Second, consider the case when $z<0$, and write $W_1(z)=D_4+D_5+D_6$, where

$$D_4=\int_{0}^{\infty}\frac{(z'-z)^{\alpha-1}\beta^\alpha e^{-\beta(z'-z)}}{\Gamma(\alpha)}\frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}}dz', \tag{8}$$

$$D_5=\int_{-\infty}^{z}\frac{(z-z')^{\alpha-1}\beta^\alpha e^{-\beta(z-z')}}{\Gamma(\alpha)}\frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}}dz', \tag{9}$$

$$D_6=\int_{z}^{0}\frac{(z'-z)^{\alpha-1}\beta^\alpha e^{-\beta(z'-z)}}{\Gamma(\alpha)}\frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}}dz'. \tag{10}$$

A closed form expression for each of D_1, D_2, …, D_6 will be derived in turn. One would like to write

$$\begin{aligned}D_1&=\int_{-\infty}^{0}\frac{(z-z')^{\alpha-1}\beta^\alpha e^{-\beta(z-z')}}{\Gamma(\alpha)}\frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}}dz'\\&=\frac{\beta^\alpha}{\Gamma(\alpha)}\left[\underbrace{\int_{-\infty}^{0}\frac{(z-z')^{\alpha-1}e^{-\beta(z-z')}}{1-e^{-4N_e(1+\frac{z'}{z})}}dz'}_{A_0}-\underbrace{\int_{-\infty}^{0}\frac{(z-z')e^{-\beta(z-z')}e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}}dz'}_{B_0}\right],\end{aligned}$$

however, this is mathematically invalid since the integrals A_0 and B_0 do not exist; for instance, A_0 contains the improper integral

$$\int_{-z-\epsilon}^{-z}\frac{1}{z+z'}dz'=-\infty$$

for small $\epsilon>0$. (Near $z'=-z$, the numerator of A_0 is integrable and positive, and the denominator expands to $\frac{4N_e}{z}(z+z')+o(z+z')$.) The technical difficulty

is that while the fixation probability $u(s)$ is analytic for all s, it can be split apart as

$$u(s) = \frac{1 - e^{-2s}}{1 - e^{-4N_e s}} = \frac{1}{1 - e^{-4N_e s}} - \frac{e^{-2s}}{1 - e^{-4N_e s}}$$

only for $s \neq 0$. Thus D_1 must be written as

$$D_1 = \int_{-\infty}^{0} = \int_{-\infty}^{-z-\epsilon} + \int_{-z-\epsilon}^{-z+\epsilon} + \int_{-z+\epsilon}^{0}$$

for some $0 < \epsilon \ll 1$. $u(s)$ can be split apart in the first and last integrals, but not the middle one, which approaches 0 as $\epsilon \to 0$. Thus, D_1 must be decomposed as

$$D_1 = \frac{\beta^\alpha}{\Gamma(\alpha)} \left[A_{0,1} + B_{0,1} + \int_{-z-\epsilon}^{-z+\epsilon} + A_{0,2} + B_{0,2} \right],$$

where the small integral $\int_{-z-\epsilon}^{-z+\epsilon}$ is either dropped or approximated numerically, and the remaining terms are given exactly by

$$\begin{aligned}
A_{0,1} &= \int_{-\infty}^{-z-\epsilon} \frac{(z-z')^{\alpha-1} e^{-\beta(z-z')} e^{4N_e(1+\frac{z'}{z})}}{e^{4N_e(1+\frac{z'}{z})} - 1} dz' \\
&= -\int_{-\infty}^{-z-\epsilon} \frac{(z-z')^{\alpha-1} e^{-\beta z + \beta z' + 4N_e + 4N_e \frac{z'}{z}}}{1 - e^{4N_e(1+\frac{z'}{z})}} dz' \\
&= -z^{\alpha-1} \int_{-\infty}^{-z-\epsilon} (1 - \frac{z'}{z})^{\alpha-1} e^{-\beta z + \beta z' + 4N_e + 4N_e \frac{z'}{z}} \sum_{t=0}^{\infty} e^{4N_e t(1+\frac{z'}{z})}\, dz' \\
&= -z^{\alpha-1} \sum_{t=0}^{\infty} \int_{-\infty}^{-z-\epsilon} \left(1 - \frac{z'}{z}\right)^{\alpha-1} e^{-\beta z + \beta z' + 4N_e + 4N_e \frac{z'}{z} + 4N_e t + 4N_e t \frac{z'}{z}} dz' \\
&= -\sum_{t=0}^{\infty} e^{8N_e + 8N_e t} z^{\alpha-1} \int_{-\infty}^{-z-\epsilon} \left(1 - \frac{z'}{z}\right)^{\alpha-1} e^{-(4N_e + 4N_e t + \beta z)(1 - \frac{z'}{z})} dz' \\
&= -\sum_{t=0}^{\infty} z^\alpha e^{8N_e + 8N_e t} \int_{2+\epsilon/z}^{\infty} s^{\alpha-1} e^{-(4N_e + 4N_e t + \beta z)s} ds \qquad (\text{with } s = 1 - \tfrac{z'}{z}) \\
&= -\sum_{t=0}^{\infty} \frac{z^\alpha e^{8N_e + 8N_e t} \gamma(\alpha, (2+\epsilon/z)(4N_e + 4N_e t + \beta z))}{(4N_e + 4N_e t + \beta z)^\alpha} \qquad (11) \\
&\quad (\text{with } x = (4N_e + 4N_e t + \beta z)s),
\end{aligned}$$

A similar derivation produces

$$\begin{aligned}
A_{0,2} = &\sum_{t=0}^{\lfloor \frac{\beta z}{4N_e} \rfloor} \frac{z^\alpha e^{-8N_e t} \hat{\gamma}(\alpha, \beta z - 4N_e t, (2-\epsilon/z)(\beta z - 4N_e t))}{(\beta z - 4N_e t)^\alpha} \\
&+ \sum_{t=\lfloor \frac{\beta z}{4N_e} \rfloor + 1}^{\infty} z^\alpha e^{-8N_e t} \sum_{k=0}^{\infty} \frac{((2-\epsilon/z)^{k+\alpha} - 1)(4N_e t - \beta z)^k}{k!(k+\alpha)}, \qquad (12)
\end{aligned}$$

$$B_{0,1} = -\sum_{t=0}^{\infty} \frac{z^{\alpha} e^{8N_e+8N_e t-4} \gamma(\alpha, (2+\epsilon/z)(4N_e+4N_e t-2+\beta z))}{(4N_e+4N_e t-2+\beta z)^{\alpha}}, \tag{13}$$

and

$$B_{0,2} = \sum_{t=0}^{\lfloor \frac{\beta z+2}{4N_e} \rfloor} \frac{z^{\alpha} e^{-4-8N_e t} \hat{\gamma}(\alpha, \beta z-2-4N_e t, 2(\beta z-2-4N_e t))}{(\beta z-2-4N_e t)^{\alpha}} + \sum_{t=\lfloor \frac{\beta z+2}{4N_e} \rfloor+1}^{\infty} z^{\alpha} e^{-4-8N_e t} \sum_{k=0}^{\infty} \frac{((2-\epsilon/z)^{k+\alpha}-1)(2+4N_e t-\beta z)^k}{k!(k+\alpha)}. \tag{14}$$

D_2, D_3, D_4, D_5, D_6 can be derived in a similar manner, and for brevity, will not be presented here. This completes the calculation of $W_1(z)$ for all $z \neq 0$ (it is assumed that the current population state is not at its optimum $z = 0$).

Given $W_1(z)$, $W_2(z) = \int_{-\infty}^{\infty} \left|\frac{z'}{z}\right| f(z')u(s)dz'$ is straightforward to compute. As before, similar to $W_1(z)$, write $W_2(z) = D_1' + D_2' + D_3'$ for $z > 0$ and $W_2(z) = D_4' + D_5' + D_6'$ for $z < 0$, where

$$\begin{aligned}
D_1' &= \int_{-\infty}^{0} \frac{\frac{-z'}{z}(z-z')^{\alpha-1}\beta^{\alpha}e^{-\beta(z-z')}}{\Gamma(\alpha)} \frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}} dz', \\
D_2' &= \int_{0}^{z} \frac{\frac{z'}{z}(z-z')^{\alpha-1}\beta^{\alpha}e^{-\beta(z-z')}}{\Gamma(\alpha)} \frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}} dz', \\
D_3' &= \int_{z}^{\infty} \frac{\frac{z'}{z}(z'-z)^{\alpha-1}\beta^{\alpha}e^{-\beta(z'-z)}}{\Gamma(\alpha)} \frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}} dz', \\
D_4' &= \int_{0}^{\infty} \frac{\frac{-z'}{z}(z'-z)^{\alpha-1}\beta^{\alpha}e^{-\beta(z'-z)}}{\Gamma(\alpha)} \frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}} dz', \\
D_5' &= \int_{-\infty}^{z} \frac{\frac{z'}{z}(z-z')^{\alpha-1}\beta^{\alpha}e^{-\beta(z-z')}}{\Gamma(\alpha)} \frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}} dz', \\
D_6' &= \int_{z}^{0} \frac{\frac{z'}{z}(z'-z)^{\alpha-1}\beta^{\alpha}e^{-\beta(z'-z)}}{\Gamma(\alpha)} \frac{1-e^{-2(1-\frac{z'}{z})}}{1-e^{-4N_e(1-\frac{z'}{z})}} dz'.
\end{aligned} \tag{15}$$

Then D_1' can be rewritten as

$$\begin{aligned}
D_1' &= \frac{1}{z}\left[\int_{-\infty}^{0} \frac{(z-z')(z-z')^{\alpha-1}\beta^{\alpha}e^{-\beta(z-z')}}{\Gamma(\alpha)} \frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}} dz' \right. \\
&\quad \left. - \int_{-\infty}^{0} \frac{z(z-z')^{\alpha-1}\beta^{\alpha}e^{-\beta(z-z')}}{\Gamma(\alpha)} \frac{1-e^{-2(1+\frac{z'}{z})}}{1-e^{-4N_e(1+\frac{z'}{z})}} dz'\right] \\
&= \frac{1}{z} D_{1,\alpha} - D_1.
\end{aligned} \tag{16}$$

Notice that the integral $D_{1,\alpha}$ has the same integrand as D_1 except for an extra factor of $|z' - z|$. The effect of this is to replace every occurrence of α by $\alpha + 1$ in the final integral formula for $D_{1,\alpha}$, except for the factor $\beta^\alpha \;/\; \Gamma(\alpha)$, which remains unchanged. This same pattern holds for all the D_i', precisely,

$$\begin{aligned}
D_1' &= \frac{1}{z} D_{1,\alpha} - D_1, \\
D_2' &= -\frac{1}{z} D_{2,\alpha} + D_2, \\
D_3' &= \frac{1}{z} D_{3,\alpha} + D_3, \\
D_4' &= -\frac{1}{z} D_{4,\alpha} - D_4, \\
D_5' &= -\frac{1}{z} D_{5,\alpha} + D_5, \\
D_6' &= \frac{1}{z} D_{6,\alpha} + D_6,
\end{aligned}$$

where the final integral formula for $D_{i,\alpha}$ differs from that for D_i as just described for $D_{1,\alpha}$ and D_1.

Finally, for $z > 0$,

$$\begin{aligned}
W(z) &= W_1(z) - W_2(z) \\
&= 2D_1 - \frac{1}{z} D_{1,\alpha} + \frac{1}{z} D_{2,\alpha} - \frac{1}{z} D_{3,\alpha},
\end{aligned} \tag{17}$$

and for $z < 0$,

$$\begin{aligned}
W(z) &= W_1(z) - W_2(z) \\
&= 2D_4 + \frac{1}{z} D_{4,\alpha} + \frac{1}{z} D_{5,\alpha} - \frac{1}{z} D_{6,\alpha}.
\end{aligned} \tag{18}$$

3 Results and Discussion

3.1 The Effect of the Distribution of Mutation Lengths

The distribution of mutation lengths in nature is unknown. However, because the gamma distribution can represent a variety of distribution shapes, employing it for the analysis covers many plausible approximations for the true distribution. In order to examine the effect of different distributions for mutation lengths, $W(z)$ is computed for different shapes to examine the effect of distributional shapes on the expected fitness changes. The simplest form is exponential, which has been used previously to approximate the distribution of the fitness effect of deleterious mutations (e.g., [7]) and rare beneficial mutations [4]. Consider first the exponential distribution, where $\alpha = 1$, and β ranges within $(0, 10]$. Shown in Figure 2, for exponential distributions with different decay rates $\beta > 0$, the expected fitness effect of the fixation of a new mutation is always positive,

suggesting that while mutations can take the population either to a state with lower fitness than the current one or a state with higher fitness, the mean fitness change will be a gain rather than a loss. In particular, for small β near zero, the expected fitness gain from the current state $z = 4$ increases with β, peaks around $\beta = 0.286$ (the expected fitness effect reaches the maximum of 0.277) and then decreases as β increases; past the peak, the larger β is, the smaller the effect of the fixation of a new mutation on the fitness change of the population. This observation is easy to understand because a large β value means that most of the mutations have a very small mutation length from the current state of the population, therefore, the fixation of the new mutation is expected to have a small effect on the population fitness change. For very small $\beta \approx 0$, both large and small mutation lengths occur with high probability, and since small mutation lengths tend to be beneficial and large mutation lengths tend to be deleterious, the effect of deleterious mutations nearly balances out the effect of beneficial mutations ($W(z) \approx 0$). Increasing β gives the smaller beneficial mutation lengths an edge, so $W(z)$ increases rapidly, until it peaks at the crossover point in the gain/loss ratio for small length mutations. This crossover occurs at the switch between prevalence of long length mutations (small β) and prevalence of short length mutations (large β).

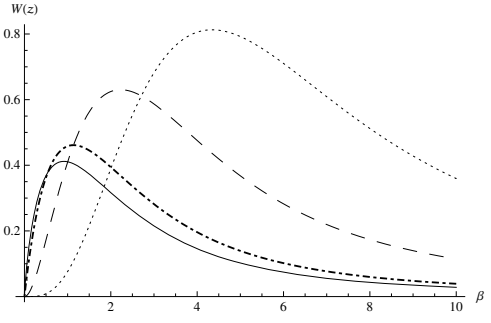


Fig. 2. The effect of the distributional shapes of mutation lengths on the expected fitness change of a new mutation with $z = 4$ and $N_e = 1000$ for all the curves, but with different α: $\alpha = 0.8$ (black), $\alpha = 1$ (dotdashed), $\alpha = 2$ (dashed), and $\alpha = 4$ (dotted)

Figure 2 also shows the effect of the distribution of mutation lengths for different αs. For small $\beta \approx 0$, the expected fitness gain due to fixation of a new mutation tends to be lower for larger α, while for $\beta \gg 1$, tends to be higher for larger α. With the current parameter settings, for example, when $\beta = 2$, the expected fitness gain is much larger for large α than for small α. In general, larger αs tend to have a wider range of β within which the expected fitness gains are large owing to the fixation of a new mutation than smaller αs. Moreover, for all different values of α and β, fixing one, there is always a maximum expected fitness gain with respect to the other, which can be obtained by setting the partial derivatives $\frac{\partial W}{\partial \alpha}$ or $\frac{\partial W}{\partial \beta}$ to zero and solving for α or β.

Previous studies have shown that large coefficients of variation in the fitness effect of both deleterious and beneficial mutations enable small populations to persist [8] and [9]. This effect is explored here by varying the coefficient of variation of mutation lengths to see what effect it has on the expected fitness change of a population. Since the coefficient of variation of a gamma distribution (with shape parameter α, scale parameter β, mean α/β, variance α/β^2) is equal to $1/\sqrt{\alpha}$, consider the relationship between $W(z)$ and α for different initial states (i.e., different z) with the same scale factor β, shown in Figure 3. Interestingly, for a specific initial state (e.g., $z = 4$), the expected fitness gain increases with α, reaches a maximum, and then approaches zero asymptotically as $\alpha \to \infty$. This shows that under the Fisher geometric adaptation model, the expected fitness gain is not a simple linear (or even monotone) function of the coefficient of variation of mutation lengths; since $\alpha \to 0$ implies the coefficient of variation $1/\sqrt{\alpha} \to \infty$, a larger coefficient of variation for mutation lengths does not necessarily lead to higher expected fitness gains. Given the definition of fitness effect $s = \frac{|z|-|z'|}{|z|}$, there might appear to be a strong correlation between the coefficient of variation of s and that of the mutation length $|z' - z|$, but the above observation indicates otherwise. Biologically, it is tempting to think that the coefficient of variation for mutation lengths should be positively correlated with the coefficient of variation for fitness effect, but one can imagine the counter-effect can also happen.

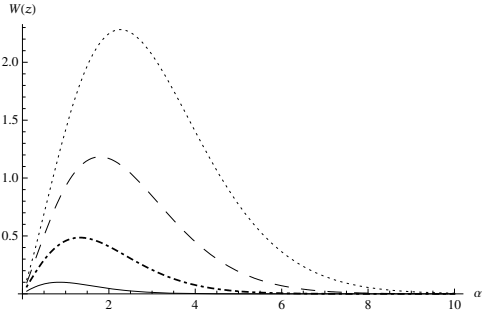


Fig. 3. The effect of the distributional shapes of mutation lengths on the expected fitness change of a new mutation with $\beta = 1$ and $N_e = 100$ for all the curves, but with different initial states: $z = 1$ (black), $z = 4$ (dotdashed), $z = 8$ (dashed), and $z = 10$ (dotted)

3.2 The Effect of the Initial State

Consider next the effect on $W(z)$ of changing the initial state z. Figure 4 shows that the starting state does affect the expected relative fitness change due to the fixation of a mutation. For the same distribution of mutation lengths, the expected fitness gain for mutation fixation increases with the distance from the "optimal" state (the origin), and approaches a constant asymptotically as $|z| \to \infty$. The asymptotic value of $W(z)$ decreases with increasing β. It was discovered,

though not shown in Figure 4, that when $\beta \leq 1$, the expected fitness gain $W(z) \to \infty$ as $z \to \infty$. This can be seen mathematically by noting that the maximum of the integrand in (4) occurs at $z' = 0$, where the exponential term $e^{|z|}$ overwhelms the exponential term $e^{-\beta|z|}$, causing the integrand (and integral) to increase without limit as $z \to \infty$. This has no biological interpretation, since distributions with $\beta < 1$ correspond to large mutation lengths $|z' - z|$ occurring with probability $\gg 0$, which is generally not true biologically.

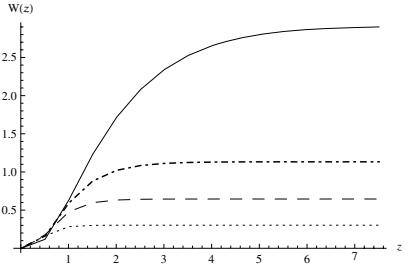


Fig. 4. The effect of the initial state on the expected fitness change of a new mutation with $\alpha = 2$ and $N_e = 100$ for all the curves, but with different β: $\beta = 2$ (black), $\beta = 3$ (dotdashed), $\beta = 4$ (dashed), and $\beta = 6$ (dotted)

3.3 The Effect of Effective Population Sizes

Population size and especially the effective population size is an important parameter in various evolutionary models, and plays an important role in determining the evolutionary trajectory of small populations and in determining the evolutionary fates of newly arising mutations. The effective population sizes of various species in nature can be difficult to measure. The mathematical derivations earlier were simplified by assuming that $N_e = N$. However, existing studies show, in several species surveyed, the effective population size (N_e) is usually much less than the census population size (N), with an estimated fraction of $N_e = 0.1N$. The derivation for $N_e \neq N$ of the analytic expression for $W(z)$ follows along the lines of the derivation for $N_e = N$ and is omitted here.

Consider the effect of N_e on the expected fitness change from one population state to another. From (3), it is clear that changing the effective population size N_e should have little effect on the expected fitness change due to the fixation of a new mutation, since unless N_e is really small, $u(s) \approx 1 - e^{-2sN_e/N}$. Changing the ratio $\frac{N_e}{N}$ has only a small effect on the final results for $\frac{N_e}{N} > 1$ (Figure 5). Therefore, it appears that under Fisher's model, the expected fitness change due to the fixation of a new mutation in a population does not depend much on the effective population size. Though mathematically explicable, it is nevertheless biologically surprising since the effective population size of a population is thought to be important in determining the fate of the population. One way to understand this is to realize that the expected fitness change can be different from one observed outcome in nature.

3.4 The Always Positive Expected Fitness Change

Since populations do go to extinction, one might expect $W(z)$ to be negative for some distribution parameters $\alpha > 0$, $\beta > 0$ and initial state z. A rigorous proof that $W(z) > 0$ for all $\alpha > 0$, $\beta > 0$, and $z \neq 0$ appears to be difficult, but there is overwhelming computational evidence that this is so. There are several possible explanations for this. One explanation is purely technical. Observe that the fixation probability $u(s)$ is strictly increasing with $u(-\infty) = 0$, $u(0) = 1/(2N)$, and $u(\infty) = 1$. Furthermore, $u(s)$ is hugely skewed in favor of beneficial mutations (fitness effect $s > 0$). For example, with $N_e = 100$, $N = 1000$,

$$u(-0.1) = 10^{-19}, \quad u(0.1) = 0.020, \qquad u(-0.5) = 10^{-88}, \quad u(0.5) = 0.095.$$

Thus, in this case, the integrand in (4) is essentially zero for $s = (|z| - |z'|)/|z| < -0.1$, which is most of the interval $-\infty < z' < \infty$, positive for $s > 0$, and negative and nonnegligible only for $-0.1 < s < 0$. Because of the shape of $u(s)$, the positive integral $\int_0^1(\cdot)ds$ is larger in magnitude than the negative integral $\int_{-0.1}^0(\cdot)ds$, giving $W(z) > 0$.

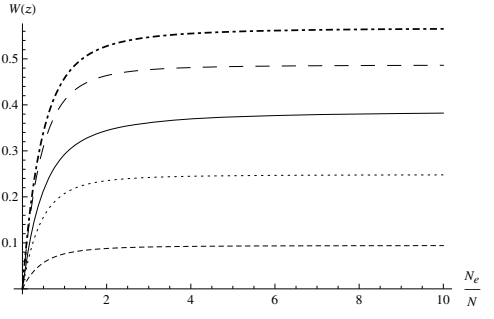


Fig. 5. The effect of the ratio $\frac{N_e}{N}$ on the expected fitness change of a new mutation with $\beta = 1$, $z = 3$, and $N_e = 1000$ for all the curves, but with different α: $\alpha = 0.5$ (black), $\alpha = 1$ (dotdashed), $\alpha = 2$ (long-dashed), $\alpha = 5$ (dotted), and $\alpha = 10$ (short-dashed). Notice the nonmonotone behavior of $W(z)$ with respect to α for a fixed $\frac{N_e}{N}$.

Another explanation recalls the definition of $W(z)$ as the *expected* fitness effect of a mutation from the initial population state z. Thus while the expected fitness effect is positive, deleterious mutations can occur and fix in the population, driving the population to extinction with positive probability—this is just not the *expected* (or average) outcome.

Another explanation is that the model here is not correct. Fitness effects may not be so simply related to mutation distances. The particular definition of fitness effect s used here may be invalid ($W(A) = e^{-|z|}$). The choice of the function representing the relationship between mutation lengths and fitness effect can influence the outcome of the model. A previous study used $W(A) = e^{-\sigma|z|^2}$ (σ is the common nonnegative intensity of selection on all traits) to define the relationship [6]. These two functions are a simplification of nature, where fitness

effect of mutation and mutation lengths can have a multitude of different relationships. Additionally, the fixation probability $u(s)$ used here may be incorrect or invalid for the particular definition of s used here. The assumed gamma distribution of mutation lengths $|z' - z|$ may not correspond to nature. While each component of the model here is an accepted model from the literature, a model is only as good as its weakest submodel or assumption.

Nevertheless, under Fisher's geometric adaptation model, the expected fitness change due to the fixation of a mutation is positive, suggesting that fixation of mutations over the long term is expected to lead to fitness gains for the population, regardless of the effective population size of the population. Thus, small populations may not necessarily be doomed due to the runaway process of fixation of deleterious mutations. It has been shown that incorporating the effect of sexual selection [7] or reverse mutations [3] into theoretical models can greatly reduce the risk of small population extinction. Also, increasing the number of dimensions that contribute to the fitness effect (pleiotropy) of mutations reduces the risk of a small population going to extinction [5]. Therefore, future work should put more emphasis on somehow measuring the fitness effect of a mutation empirically and understanding how the fitness effect of a mutation is determined by the interaction of different genetic components of a population in order to better model the risk of population extinction.

Acknowledgments

This work was supported in part by NSF Grants DMI-0355391 and IIS-0710945.

References

1. Crow, J.F., Kimura, M.: An Introduction to Population Genetics Theory. Harper Row, New York (1970)
2. Fisher, R.A.: Genetical Theory of Natural Selection. The Clarendon Press, Oxford (1930)
3. Lande, R.: Risk of population extinction from fixation of deleterious and reverse mutations. Genetica 102, 21–27 (1998)
4. Orr, H.A.: The distribution of fitness effects among beneficial mutations in Fisher's geometric model of adaptation. J. Theor. Biol. 238, 279–285 (2006)
5. Poon, A., Otto, S.P.: Compensating for our load of mutations: freezing the meltdown of small populations. Evolution Int. J. Org. Evolution 54, 1467–1479 (2000)
6. Waxman, D., Welch, J.J.: Fisher's microscope and Haldane's ellipse. Am. Nat. 166, 447–457 (2005)
7. Whitlock, M.C.: Fixation of new alleles and the extinction of small populations: drift load, beneficial alleles, and sexual selection. Evolution Int. J. Org. Evolution 54, 1855–1861 (2000)
8. Zhang, L., Watson, L.T.: Note on the computation of critical effective population sizes. J. Comput. Biol. 14, 950–960 (2007)
9. Zhang, L., Watson, L.T.: Analysis of the fitness effect of compensatory mutations. HFSP J. 3, 47–54 (2009)

Author Index