ABSTRACT

Title of dissertation: COMPUTATIONAL METHODS IN PROTEIN STRUCTURE COMPARISON AND ANALYSIS OF PROTEIN INTERACTION NETWORKS

Elena Zotenko
Doctor of Philosophy, 2007

Dissertation directed by: Professor Dianne P. O'Leary
Department of Computer Science
Doctor Teresa M. Przytycka
NCBI/NLM/NIH

Proteins are versatile biological macromolecules that perform numerous functions in a living organism. For example, proteins catalyze chemical reactions, store and transport various small molecules, and are involved in transmitting nerve signals. As the number of completely sequenced genomes grows, we are faced with the important but daunting task of assigning function to proteins encoded by newly sequenced genomes. In this thesis we contribute to this effort by developing computational methods for which one use is to facilitate protein function assignment.

Functional annotation of a newly discovered protein can often be transferred from that of evolutionarily related proteins of known function. However, distantly related proteins can still only be detected by the most accurate protein structure alignment methods. As these methods are computationally expensive, they are combined with less accurate but fast methods to allow large-scale comparative studies. In this thesis we propose a general framework to define a family of protein structure comparison methods that reduce protein structure comparison to distance computation between high-dimensional vectors

and therefore are extremely fast.

Interactions among proteins can be detected through the use of several mature experimental techniques. These interactions are routinely represented by a graph, called a protein interaction network, with nodes representing the proteins and edges representing the interactions between the proteins. In this thesis we present two computational studies that explore the connection between the topology of protein interaction networks and protein biological function.

Unfortunately, protein interaction networks do not explicitly capture an important aspect of protein interactions, their dynamic nature. In this thesis, we present an automatic method that relies on graph theoretic tools for chordal and cograph graph families to extract dynamic properties of protein interactions from the network topology.

An intriguing question in the analysis of biological networks is whether biological characteristics of a protein, such as essentiality, can be explained by its placement in the network. In this thesis we analyze protein interaction networks for Saccharomyces cerevisiae to identify the main topological determinant of essentiality and to provide a biological explanation for the connection between the network topology and essentiality.

COMPUTATIONAL METHODS IN PROTEIN STRUCTURE COMPARISON AND ANALYSIS OF PROTEIN INTERACTION NETWORKS

by

Elena Zotenko

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2007

Advisory Committee:
Professor Dianne P. O'Leary, Co-Chair
Doctor Teresa M. Przytycka, Co-Chair
Professor Samir Khuller
Associate Professor Stephen M. Mount
Assistant Professor Mihai Pop

DEDICATION

to my mother Ludmila

ACKNOWLEDGMENTS

I would like to use this opportunity to express my gratitude to all the people who made my graduate years a truly exciting and enriching experience. At an academic level, I had the opportunity to interact with and learn from world-class scholars. At a personal level, I met wonderful people who were fun to be with and provided support and encouragement during challenging moments.

I am greatly indebted to my advisors, Dianne O'Leary and Teresa Przytycka, who were a source of constant inspiration and motivation. Without their guidance and support this thesis would not be possible.

I would like to thank the members of my advisory committee, Samir Khuller, Steve Mount and Mihai Pop, for their time and valuable feedback.

Several chapters of this thesis are based on published papers, therefore I would like to acknowledge my co-authors, Katia Guimaraes, Rezarta Islamaj-Dogan, Raja Jothi, Julian Mestre, and John Wilbur. Rezarta and John contributed useful discussions on the usage of machine learning strategies at the early stages of the project described in Section 3.2.4.2. Katia and Raja were involved in the project described in Chapter 5 and provided useful discussions and contributed greatly to improving the presentation of the original paper. Julian Mestre was involved in designing a fast algorithm described in Section 6.2.2.

I would like to acknowledge the Intramural Research Program of the NIH, National Library of Medicine, through which I was financially supported during the last four years of my graduate studies.

I am grateful to the faculty of the Computer Science Department, Bobby Bhattacharjee, Samir Khuller, Dianne O'Leary, David Mount, James Reggia, and Aravind Srinivasan for offering enlightening and thought provoking graduate level courses. I am also grateful to the department staff members, Gwen Kaye, Fatima Bangura and Jennifer Story, who made dealing with the administrative issues as painless as possible.

I was fortunate to be a member of Teresa Przytycka's research lab. I would like to thank the other lab members, Raja Jothi, Katia Guimaraes, and Jie Zheng, for their support and insightful feedback on many aspects of my research work.

Finally, I would like to thank my mother for her constant support and encouragement and my husband and daughter who brought a special meaning to my life.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Proteins are versatile biological macromolecules that perform numerous functions in a living organism. For example, proteins catalyze chemical reactions, store and transport various small molecules, and are involved in transmitting nerve signals [55].

It is common to distinguish between *molecular function* and *cellular function* of a protein. The molecular function denotes protein chemical/physical activity at the molecular level. Cellular function, on the other hand, denotes protein activity at the cellular level such as involvement in a particular signaling or metabolic pathway. For example, at the molecular level the enzyme *DNA polymerase* creates a copy of DNA during cell division. At the cellular level the enzyme is involved with many other proteins in a complex process of duplicating the cell's genome during every cell division [12].

As the number of completely sequenced genomes grows (there are nearly 600 completely sequenced genomes available at the NCBI website) we are faced with the important but daunting task of assigning function to proteins encoded by newly sequenced genomes. In this thesis we contribute to this effort by developing computational methods for which one use is to facilitate protein function assignment at the molecular and cellular levels.

Homologous proteins are proteins that descend from a common ancestor. It is widely believed that homologous proteins perform similar functions in different organisms. In fact, one of the oldest and most powerful approaches to infer protein function of newly discovered proteins relies on using homology relationships to assign function. During

evolution, protein structure is more conserved than protein sequence, so the homology relationship between distantly related proteins can only be detected by protein structure alignment methods.

Over the years, many reliable protein structure alignment methods were proposed [94, 95, 65, 50, 48, 111], but due to the inherent difficulty of the protein structure alignment problem these methods are computationally expensive and therefore cannot be used in large-scale comparative studies of protein structure. To overcome this deficiency, a reliable method is usually combined with a less accurate but fast protein structure comparison method. In this thesis we propose a general framework, the *structural footprinting framework*, that defines a family of fast protein structure comparison methods. The framework can be used to design a variety of methods that allow extremely fast and simple protein structure comparison. We present an extensive experimental evaluation to assess the potential of our framework in designing fast protein structure comparison methods.

The complexity in biological systems arises not only from various individual protein molecules but also from their organization into systems with numerous interacting partners. Over the past decade several high-throughput experimental techniques to detect protein interactions were developed [38, 102, 112]. These experimentally-determined interactions are routinely represented by a graph, a *protein interaction network*, with nodes representing the proteins and edges representing the interactions between the proteins. The study of the topological properties of these networks has become an important tool in studying protein function at the cellular level and formulating hypotheses about the general organization principles of biological systems. In this thesis we present two computational studies to explore the connection between the topology of protein interaction networks and protein biological function.

Some cellular processes proceed through an orderly formation of multi-protein complexes. An example of such cellular process is the eukaryotic ribosome assembly pathway [35], which involves in addition to four RNAs and around 80 ribosomal proteins nearly 200 auxiliary proteins that are not part of the mature ribosomes. The ribosome assembly is believed to proceed in a highly coordinated manner, where the participating proteins join and leave the pathway in a fixed order and this order is critical for the proper ribosome assembly. Even though the major components of the pathway and their interactions are known, there is little knowledge about the dynamical properties of the pathway, in particular about the order in which the multi-protein complexes are formed. In this thesis we present an automatic method that elucidates the temporal order of complex formation during a cellular process from the topology of the protein interaction network spanning the process components.

A systematic gene deletion screen in the yeast *Saccharomyces cerevisiae* revealed that about 18% of the genes are essential for growth on rich glucose medium [49], meaning cells lacking any one of these genes are not viable. An intriguing question in the analysis of biological networks is whether biological characteristics of a protein, such as essentiality, can be explained by its placement in the network, i.e., whether topological prominence implies biological importance. One of the first connections between the two in the context of a protein interaction network, the so-called *centrality-lethality rule*, was observed by Jeong and colleagues [73] who demonstrated that high-degree nodes or *hubs* in a protein interaction network of *Saccharomyces cerevisiae* contain more essential proteins than would be expected by chance. Since then the correlation between degree and essentiality was confirmed by other studies [120, 58, 9, 121], but until recently [62] there was no systematic attempt to examine the reasons for this correlation. In particular, what is the

main topological determinant of essentiality? Is it the number of immediate neighbors or some other, more global topological property that essential proteins may have in a protein interaction network?

To identify the main topological determinant of essentiality and to provide a biological explanation for the connection between the network topology and essentiality, we perform a rigorous analysis of five genome-wide protein interaction networks for *Saccharomyces cerevisiae*. We demonstrate that the majority of hubs are essential due to their involvement in *essential complex biological modules*, a group of densely connected proteins, with shared biological function, that are enriched in essential proteins. Moreover, we reject two previously proposed explanations for the centrality-lethality rule, one relying on the assumption that essential hubs maintain the overall network connectivity and another relying on the recently published essential protein interactions model.

The rest of this chapter gives a high-level description of our contributions.

1.1 Structural Footprinting in Fast Protein Structure Comparison

Among the growing number of different approaches to speed up protein structure comparison, *projection methods* offer a promising new solution to the problem by first mapping a protein structure to a high-dimensional vector. Once the mapping is done, protein structural similarity is approximated by the distance between the corresponding vectors. The projection method's approach to protein structure comparison is schematically shown in Figure 1.1.

By reducing structural comparison to distance computation between vectors, projection methods achieve a considerable speed-up over full-fledged protein structure alignment methods. (Once vector representations are computed, it takes on average 500 seconds for a

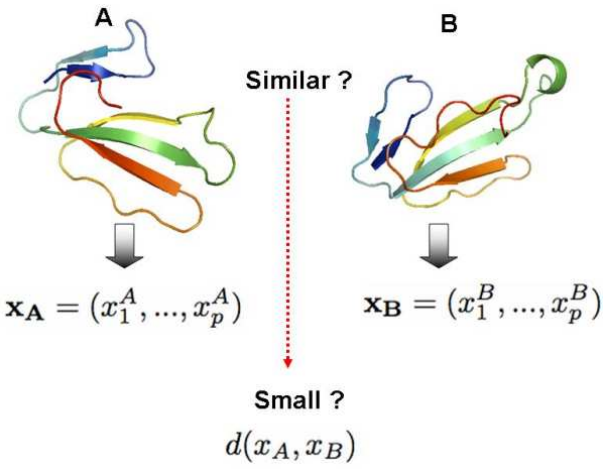


Figure 1.1: To compare structures A and B, a projection method will first map them to a vector in a high-dimensional vector space. Thus, structure A is mapped to vector $\vec{x}_A$ and structure B, to vector $\vec{x}_B$. The structure comparison is then reduced to the distance computation between these vectors; i.e., the structures are similar if distance $d(\vec{x}_A, \vec{x}_B)$ is small.

projection method to perform all pairwise comparisons among $5,024$ structures. Compare this to nearly five months it would take DALI [65], a highly accurate protein structure alignment method, to perform the same number of pairwise comparisons.) Therefore, projection methods can be combined with more accurate full-fledged methods to allow high-throughput comparative structure analysis. Protein structure alignment servers are routinely used to compare a query protein structure against a large database of structures such as the set of sequence non-redundant protein domains in the CATH database [96], which currently contains $7,794$ structures. A projection method can be used to rank the structures in the database, allowing the more computationally expensive residue-based structure alignment method to be applied only to the highest ranked (small) fraction of the database. Furthermore, a vector representation of protein structure produced by a projection approach can be combined with machine learning algorithms to provide powerful classification schemes, indexing algorithms to provide fast retrieval of similar structures, clustering and dimension reduction algorithms to provide a compact representation of

the protein structure universe such as in [68], etc. Therefore, improving the performance of projection methods and understanding the limits of these techniques is particularly important.

The central question in the projection method approach to protein structure comparison is how to devise a mapping that is able to capture all the salient features of protein structure. Currently known projection methods [22, 44, 103, 23] employ very different approaches to the mapping construction. In particular, the mapping should be able to tolerate structural variability that is characteristic of distantly related protein structures.

In this thesis we adopt the high-level idea behind the LFF projection method [23] to define a general framework, which we call the *structural footprinting framework*, for designing projection methods. In fact, the same high-level idea is common to diverse application areas, such as text mining [83] and classification of biological networks [89], in which a complex object is represented as a high-dimensional vector of counts or *footprint* of its small size motifs. In the case of the structural footprinting framework, such motifs correspond to structural fragments. Since the space of all such fragments is not discrete, a finite set of representative structural fragments or *models* is selected. The set of models can be thought of as a *structural alphabet* used to describe the protein structure. The framework does not define the specification and representation of structural fragments. Thus, any two structural footprinting methods differ in the set of models they use; in fact a large number of methods can be generated by varying the type of structural fragments used and the amount of detail in their representation.

The main objective of our study is to explore the potential of structural footprinting in fast protein structure comparison and to understand the influence of the structural alphabet used by a structural footprinting method on its performance. To address the

first point we propose the Secondary Structure Element Footprint (SSEF) method that uses a structural alphabet derived from regions of the structure that are believed to be most conserved during evolution. We present an extensive evaluation of the database retrieval ability of the SSEF method as compared to established protein structure/sequence alignment methods and other projection methods. The results of our evaluation indicate that the structural footprinting framework can be used to produce projection methods that not only outperform other projection methods but also compare favorably with some full-fledged protein structure alignment methods.

To address the influence of the structural alphabet, we propose another structural footprinting method, the SEGment Footprint (SEGF) method, that together with the SSEF and LFF methods samples a variety of structural alphabets. We present a comprehensive evaluation of these methods based on their ability to detect structural similarity characteristic of evolutionarily related structures. Our experiments indicate that no single method performs the best in all cases. To take advantage of the relative strengths of the methods we propose strategies to combine the methods to achieve better performance.

1.2 Dynamic Formation of Multiprotein Complexes

Recent proteomic studies characterized interactions among the components of many cellular processes [6, 19]. Moreover, genome-wide interaction maps exist for several model organisms [118, 70, 80, 46, 51, 82, 106]. Can the readily available protein interaction data be used to elucidate the dynamical properties of cellular processes? In particular, can we infer something about the temporal order of multi-protein complex formation during a cellular process from the topology of the underlying protein interaction network?

Even though protein interaction networks do no explicitly capture the dynamic na-

ture of protein interactions, there are graph theoretic tools that under certain assumptions allow the extraction of this information from the topology of the network. Unfortunately, research attempts in this direction are very limited. In fact, we are aware of only one method, the method due to Farach-Colton *et al.* [34], which takes advantage of *interval graph theory* to reason about the order in which auxiliary proteins enter and leave the ribosome assembly pathway.

In this thesis we develop an automatic method, the *Complex Overlap Decomposition* (COD) method, to elucidate the order of multi-protein complex formation during a cellular process from the topology of the corresponding protein interaction network. Our method relies heavily on the graph theoretic results for *chordal* and *cograph* graph families. Given a protein interaction network spanning the process components, our method identifies protein complexes and produces a *Tree of Complexes* representation. A Tree of Complexes is a tree whose nodes are protein complexes and whose topology satisfies certain continuity constraints; namely complexes that share a protein must be connected. In this way, our representation captures the manner in which proteins enter and leave the complexes and therefore can be used to hypothesize about the order of their formation. Indeed, once the root of the tree is fixed, the representation induces a partial order on the complexes, which in turn can be used to infer temporal relationships.

We apply the COD method to two protein interaction networks underlying well studied cellular processes in *Saccharomyces cerevisiae* (bakers yeast): the mating pheromone signaling pathway and the DNA replication module. Our results show that the COD method gives insight into the analysis of protein interaction networks.

1.3 Topological Determinants of Lethality

In their paper, Jeong and colleagues [73] suggested that over-representation of essential proteins among high-degree nodes can be attributed to the central role hubs play in mediating interactions among numerous, less connected proteins. Indeed, the removal of hubs disrupts the connectivity of the network, as measured by the network diameter or the size of the largest connected component, more than the removal of an equivalent number of random nodes [1, 73]. Therefore, under the assumption that the organism's function depends on the connectivity among various parts of its interactome, hubs are predominantly essential because they play a central role in maintaining this connectivity.

Recently, He and colleagues challenged the hypothesis of essentiality being a function of a global network structure and proposed that the majority of proteins are essential due to their involvement in one or more *essential protein interactions* that are distributed uniformly at random among the network edges [62]. Under this hypothesis, hubs are predominantly essential because they are involved in more interactions and thus are more likely to be involved in one which is essential.

In this thesis we carefully evaluate each of the proposed explanations for the centrality-lethality rule using five genome-wide protein interaction networks for *Saccharomyces cerevisiae* compiled from diverse sources of interaction evidence [29, 101, 10, 25, 72]. In addition to degree, we consider several other measures of topological prominence, some of which are influenced more by the global structure of the network, and find that degree is a better predictor of essentiality than any other measure tested. On the other hand, we observe that the hypothesis proposed by He and colleagues [62] does not hold in the tested networks. Most notably, the assignment of essentiality through uniform distribution of essential protein interactions among the edges of the network fails to reproduce

basic clustering patterns of essential proteins observed in the real data.

Motivated by our findings, we propose an alternative explanation for the centrality-lethality rule which is based on the existence of *essential complex biological modules*. Essential complex biological modules, abbreviated here as ECOBIMs, are biological processes that are: (i) indispensable for organism's vitality; (ii) composed of proteins that interact with each other in a dense pattern of protein interactions. It is reasonable to assume that members of ECOBIMs are predominantly essential as they are involved in vital biological processes and are difficult to substitute for due to complexity of their protein-protein interactions.

It should be noted that the existence of ECOBIMs is well documented. For example, the MIPS database of manually curated multi-protein complexes [88] contains several large multi-protein complexes, such as proteasome or cytoplasmic ribosomal subunits, whose components are mostly essential. There are also processes that involve several interacting multi-protein complexes, such as RNA Polymerase II general transcriptional machinery [59] or ribosome biogenesis and assembly [35, 42]. Moreover, essential proteins are not distributed evenly among the MIPS complexes; i.e., there are complexes whose components are mostly essential, and there are complexes whose components are mostly non-essential. The same phenomenon was recently observed in the set of automatically identified protein complexes [60].

We hypothesize that in the tested networks the majority of the hubs are members of ECOBIMs, and since ECOBIMs are enriched in essential proteins, so are the participating hubs. To test our hypothesis we develop two complementary methods to extract putative ECOBIMs from a protein interaction network. Both methods use GO annotation [7] and a set of 192 manually derived biological process GO terms [92] to delineate the boundaries

of biological processes. For each tested network, we demonstrate that the set of putative ECOBIMs identified by our methods explains the enrichment of high-degree nodes in essential proteins. In particular, the majority of essential hubs belong to one or more ECOBIMs. Moreover, the fraction of essential proteins among hubs that are not members of ECOBIMs is significantly lower than the fraction of essential proteins among the nodes of the network, so that non-ECOBIM hubs are depleted in essential proteins.

1.4 Outline of the Thesis

The remainder of this thesis is organized as follows. In Chapter 2 we provide relevant background information on protein structure and protein structure alignment. Chapter 3 describes our contributions to fast protein structure comparison. In Chapter 4 we review the experimental techniques used to characterize protein interactions and describe the protein interaction networks used in this thesis. Chapter 5 outlines our methodology on using the network topology to infer the order of dynamic complex formation during a cellular process. Our study of the connection between network topology and gene essentiality is described in Chapter 6. Finally, in Chapter 7 we summarize our contributions and present directions for future work.

Chapter 2

Protein Structure Preliminaries

Every naturally occurring protein is able to fold to a specific three-dimensional shape, which is closely related to the ability of the protein to perform its biological function. Moreover, it is widely accepted that protein structure is much more conserved during evolution than protein sequence [24, 105, 109]. Therefore protein structure alignment is an important tool for understanding principles of protein function and evolution.

The purpose of this chapter is to introduce the reader to relevant background information on protein structure and protein structure alignment. We start with a brief description of protein structure in Section 2.1 and protein structure repositories and classification databases in Section 2.2. We then give a formal definition of the protein structure alignment problem in Section 2.3.

2.1 Principles of Protein Structure

A protein chain is a sequence of amino acids linked together by peptide bonds (cf. Figure 2.1). All twenty standard amino acids share a common template structure, which consists of a central carbon atom (C_α) with an attached hydrogen atom, an amino group, and a carboxyl group. What distinguishes one amino acid from another is the particular *side chain*, called the residue, also attached to the central carbon atom. During protein synthesis the carboxyl group of one amino acid binds to the amino group of the next amino acid, forming a *peptide bond*. When many amino acids are bound together by

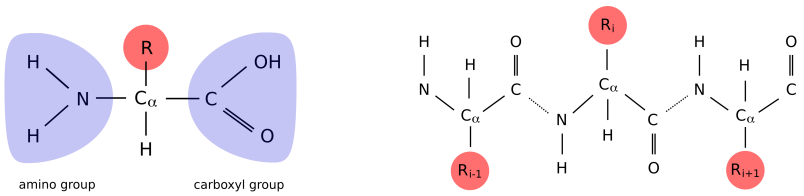


Figure 2.1: Amino acids are the basic building blocks of proteins. **(a)** Amino acid. A central carbon atom C_α is attached to an amino group NH_2, a carboxyl group $COOH$, a hydrogen atom H and a side chain R. **(b)** Segment of polypeptide chain with three residues R_{i-1}, R_i and R_{i+1}.

peptide bonds they form a *polypeptide chain* or *backbone* from which various side chains project.

The molecular forces between the atoms of a protein and their environment drive the folding of the polypeptide chain to a unique three-dimensional structure or *native state*. A protein's native state not only maximizes its stability but also places functional residues at accessible spatial locations, allowing the protein to bind other proteins and molecules. Thus, there is a close connection between the structure of a protein and its ability to function.

It is common to distinguish between different levels of protein structure. The amino acid sequence of the protein's polypeptide chain is called its *primary structure*. Certain segments of the polypeptide chain form regular substructures or *secondary structure elements* (SSEs): α-helices and β-strands. The sequence of secondary structure elements is called the *secondary structure*. The *tertiary structure* is the three-dimensional shape of a protein chain. The protein may contain several chains, forming its *quaternary structure*. A protein *domain* is a segment or several segments of the protein backbone that form a compact globular substructure, is believed to be an autonomous folding unit, carries a specific biological function, and recurs as a substructure in different proteins. It is widely

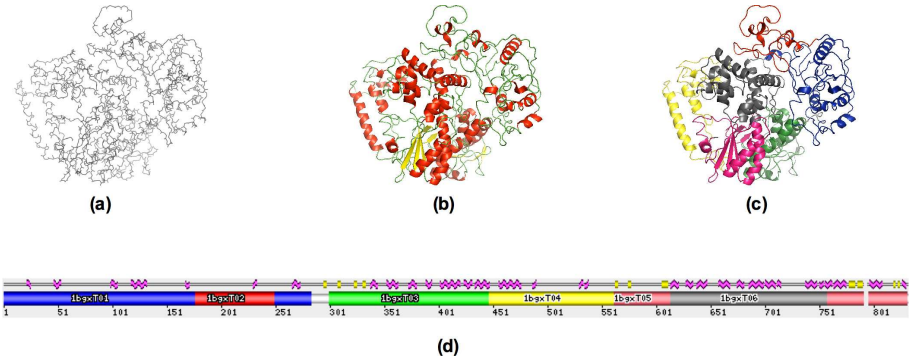


Figure 2.2: The levels of protein structure are exemplified on the enzyme DNA polymerase I from the bacterium *Thermophilus aquaticus* (PDB code `1bgx`). **(a)** The three-dimensional shape of the polypeptide backbone. **(b)** The secondary structure assignment is shown by coloring the segments of the backbone which correspond to β-strands with yellow, α-helices with red, and regions in between the secondary structure elements with green. **(c)-(d)** The protein domains, six in total, are shown on the three-dimensional structure in (c) and along the primary sequence in (d).

accepted that a protein domain, rather than a protein chain, is an elementary unit of protein structure and evolution.

Figure 2.2 exemplifies the concepts discussed above on the structure of the enzyme DNA polymerase I, from the bacterium *Thermophilus aquaticus*, that is part of a large molecular machinery that performs DNA replication. The enzyme is a big protein made up of 828 aminino acids (the polypeptide chain is shown in Figure 2.2(a)), 11 β-strands and 42 α-helices (the secondary structure assignment is shown in Figure 2.2(b) with strands colored yellow, helices colored red, and regions of the backbone in between the secondary structure elements colored green). The enzyme is made of 6 domains, `1bgxT01-1bgxT06`, whose position along the primary sequence is schematically shown in Figure 2.2(d) and on the three-dimensional structure is shown in Figure 2.2(c).

It has been established that some elements of protein structure are more conserved during evolution than others. In particular, secondary structure elements that are important factors in stabilizing the protein's three-dimensional structure are more conserved

than loop regions, regions of the backbone in between the secondary structure elements.

2.2 Protein Structure Repositories and Classification Databases

The Protein Data Bank (PDB) [14] is the most comprehensive repository of structure data for biological macromolecules. Among other things, this repository contains primary structure information, secondary structure information, and atomic coordinates of a protein structure.

As of July 2007, PDB contains structure data for 41,095 proteins. On average, proteins have between 100 and 300 residues. There are big proteins that contain 1000 or more residues and small proteins that contain at most 30 residues. The number of SSEs is on average between 3 and 20. Once again there is a large variation; there are structures that have 50 or more SSEs and structures that do not have SSEs at all.

The extensive growth of protein sequence and structure information has resulted in the creation of numerous classification resources for organizing proteins [100]. Two main structure-based classification databases, SCOP [91] and CATH [96], combine sequence, structural and functional information to provide a hierarchical classification of known protein domains in the PDB.

In the CATH database, for example, protein domains are organized into a four-level hierarchy [98]: class, architecture, topology and homologous superfamily.

- **Homologous superfamily**. Members of the same **homologous superfamily** group share a clear common evolutionary origin supported either by significant sequence similarity or significant structural and functional similarity.

- **Topology**. The homologous superfamilies are grouped into topologies, where members of the same topology group share significant structural similarity but are not

required to share sequence or functional similarity necessary to infer a common evolutionary origin.

- **Architecture**. The architecture level groups proteins based on coarse topological organization of secondary structure elements.

- **Class**. Finally the class level groups proteins according to secondary structure element content: mainly α, mainly β, mixed α and β, or small structures.

As evolutionary changes accumulate, the protein's sequence and its three-dimensional structure change. It is common to quantify the amount of sequence divergence between two proteins by the fraction of identical residues in their sequence alignment. For example, 20% sequence identity means that 20% of the residues of the smallest protein are aligned to identical residues in the larger protein. It is well known that conventional protein sequence alignment methods such as BLAST [4] fail to compute correct alignments when protein sequence identity drops below $40\% - 30\%$ [31]. Both CATH and SCOP classification databases provide a set of sequence non-redundant proteins domains in the database; the CATH database uses a 35% identity threshold to produce this data set, whereas the SCOP database uses a 40% sequence identity threshold.

2.3 Protein Structure Alignment Problem

As with sequence alignment, structural alignment involves detection of a set of equivalent residues that optimize a given similarity score. The similarity score measures the *amount of similarity* between two protein structures and is usually a function of two interdependent factors: the number of aligned residues and how well the aligned residues can be superimposed by a rigid body transformation.

We will now give a more formal definition of an optimization problem involved in protein structural alignment. For the purpose of the alignment, a residue is represented by a point in $3D$, usually the atomic coordinate of its C_α atom. A protein structure is represented by an ordered set of points, $q = \{q_1, ..., q_n\}$, which corresponds to its residues. An alignment between two structures, q and p, is a one-to-one mapping between a subset $\hat{q}$ of residues in q and a subset $\hat{p}$ of residues in p, where the mapping is defined by a one-to-one function $\phi : \hat{q} \rightarrow \hat{p}$. An optimal alignment is one that maximizes a similarity score objective function $S(\hat{q}, \phi(\hat{q}))$. Different methods optimize different objective functions. For example, the widely used DALI method [65] seeks to minimize a variant of the following objective function:

$$S(\hat{q}, \phi(\hat{q})) = \sum_{q_i \in \hat{q}} \sum_{q_j \in \hat{q}} A - |d(q_i, q_j) - d(\phi(q_i), \phi(q_j))|$$

where $d(q_i, q_j)$ denotes the Euclidean distance between points q_i and q_j. The balance between two interdependent factors mentioned above is realized through the terms A and $|d(q_i, q_j) - d(\phi(q_i), \phi(q_j))|$. Each aligned pair of residues contributes a constant factor, A, to the similarity score, which is penalized by the amount of structural deviation in the pair's spatial orientation, $|d(q_i, q_j) - d(\phi(q_i), \phi(q_j))|$.

Structural alignment is a difficult problem and the majority of optimization problems involved are either NP-hard or high degree polynomial [78]. Moreover, even heuristics employed by structural alignment methods become prohibitively expensive when an extensive comparison of a query structure to a large database of structures, such as the PDB, needs to be carried out.

Chapter 3

Fast Protein Structure Comparison with Structural Footprinting [1]

Protein structure comparison is an important tool that helps biologists understand various aspects of protein function and evolution. Unfortunately, protein structure alignment is a difficult problem and highly accurate protein structure alignment methods are computationally expensive. In fact, the execution time of these methods becomes prohibitively expensive for large-scale comparative structure analysis, such as when a query protein structure needs to be compared to a large database on a regular basis. (For example, it would take the DALI program, one of the most accurate protein structure alignment methods, nearly five months to perform all pair-wise comparisons of $5,024$ domains.)

Over the years, numerous methods for speeding up protein structure alignment were developed [66, 2, 84, 22, 44, 104, 23]. One of the recently pursued approaches is the so-called *projection approach*, where a protein structure is mapped to a high-dimensional vector and structural similarity is approximated by distance between the corresponding vectors. Methods that employ this approach include PRIDE [22, 44], SGM [103], and LFF [23]. In PRIDE [22, 44], Carugo *et al.* compute all pairwise distances between the central carbon atoms k residues apart (k ranging between three and thirty), and use the

[1] This chapter is derived from "Secondary structure spatial conformation footprint: A novel method for fast protein structure comparison" by E. Zotenko, D. P. O'Leary and T. M. Przytycka, BMC Structural Biology, 6:12, 2006, and "Structural footprinting in protein structure comparison: The impact of structural fragments" by E. Zotenko, R. Islamaj Dogan, W. J. Wilbur, D. P. O'Leary, and T. M. Przytycka, BMC Structural Biology, 7:53, 2007.

distance distributions as a descriptor of protein structure. In SGM [103], Rogen *et al.* map a protein backbone into R^{30} using geometric invariants borrowed from Knot Theory. In LFF [23], Choi *et al.* apply an idea common to diverse application areas, including text mining [83] and classification of biological networks [89], in which a complex object is represented as a high dimensional vector of counts or *footprint* of its small size motifs. In the case of protein structure, such motifs correspond to structural fragments. Choi *et al.* use pairs of backbone segments of size ten as structural fragments. Since the space of all such fragments is not discrete, a finite set of representative structural fragments or *models* is selected. Given a protein structure, its structural footprint is computed by making each structural fragment in the structure contribute a count of one to the closest (most similar) model.

In this thesis we adopted the idea behind the LFF method to define a framework, which we call *structural footprinting framework*, for designing projection methods. The framework predetermines certain steps taken to create a vector representation of a protein structure. Thus, any structural footprinting method first selects a representative set of structural fragments or models. The set of models can be thought of as a *structural alphabet* used by the method to describe protein structure. Once the models are selected, the method maps a protein structure to a vector in which each dimension corresponds to a particular model and "counts" the number of times the model appears in the structure. However, the framework leaves the specification and representation of structural fragments to the structural footprinting method at hand. Thus, any two structural footprinting methods differ in the set of models they use; in fact a large number of methods can be generated by varying the type of structural fragments used and the amount of detail in their representation. The structural footprinting framework is described in detail in

Section 3.1.1.

The main objective of our study was to explore the potential of the structural footprinting framework as applied to fast protein structure comparison. Toward this end, we developed two novel structural footprinting methods, the Secondary Structure Element Footprint (SSEF) and SEGment Footprint (SEGF) methods, that together with the LFF method sample a variety of structural alphabets. The SSEF and SEGF methods are described in Sections 3.1.2 and 3.1.3 respectively. We then performed an extensive experimental evaluation to assess the performance of structural footprinting methods as compared to: (i) well established protein structure and sequence alignment methods, described in Section 3.2.2, and (ii) other projection methods, described in Section 3.2.3. We also explored how the performance of a structural footprinting method depends on the structural alphabet used by the method and whether the SSEF, SEGF, and LFF methods can be combined to achieve a better performance. The proposed strategies to combine these methods and the results of experimental evaluation are described in Section 3.2.4.

3.1 Two Novel Structural Footprinting Methods

3.1.1 General Algorithmic Framework

Structural footprinting methods are a family of projection methods that use the same general algorithmic framework to produce a vector representation of protein structure: (i) select a representative set of structural fragments or *models*, (ii) map a protein structure to a vector in which each each dimension corresponds to a particular model and "counts" the number of times the model appears in the structure. While the model selection step is performed only once, the second step is performed every time a protein structure needs to be mapped to the structural footprint.

The framework leaves the specification of structural fragments and their representation to the structural footprinting method at hand. Therefore, every structural footprinting method has to support two operations: (i) extract all the structural fragments present in a given protein structure and (ii) measure the similarity between a pair of structural fragments.

The models should provide an adequate coverage of structural fragments present in the protein structure universe, i.e., every structural fragment should be close enough to at least one model. To achieve this goal, structural fragments are extracted from a representative set of protein structures and then clustered using the k-means clustering algorithm [71]. The resulting cluster centers are output as models.

Let us denote by $\mathcal{M} = \{m_1, ..., m_p\}$ the set of models selected in the model selection step. The footprint of a structure Q is a vector in R^p, denoted by $\vec{f}_Q$, where each dimension corresponds to a specific model and its value is equal to the score accumulated by the model over all structural fragments in Q. We allow a structural fragment to contribute to several models, where the amount of contribution is inversely proportional to the distance between the fragment and the model; the contributions are normalized to sum up to one. A footprint $\vec{f}_Q$ is formally defined as follows.

$$
\begin{aligned}
\vec{f_Q} &= (f_1^Q, ..., f_p^Q) \\
f_i^Q &= \sum_{s:d(s,m_i)<\gamma} c(s, m_i) \\
c(s, m_i) &= \frac{\exp(-d(s, m_i)^2/a)}{\sum_{m_j} \exp(-d(s, m_j)^2/a)}
\end{aligned}
$$

s is a structural fragment of Q

$c(s, m_i)$ is a contribution of s to model m_i

$d(s, m_i)$ is the distance between s and a model m_i

a is a scale factor

γ is a threshold

A structural fragment s contributes to a model m only if they are similar enough, i.e., the distance $d(s, m)$ is below a certain threshold γ. The value of this threshold and the scale factor a are determined from the distribution of distances of structural fragments to the closest model observed in the protein structure universe.

Once footprints are computed, the structural similarity between two protein domains is measured by the Pearson correlation coefficient of their footprints $\vec{f_Q}$ and $\vec{f_P}$:

$$
\frac{\sum_{i=1}^{p} (f_i^Q - \mu_Q)(f_i^P - \mu_P)}{\sqrt{\sum_{i=1}^{p} (f_i^Q - \mu_Q)^2} \sqrt{\sum_{i=1}^{p} (f_i^P - \mu_P)^2}}
$$

where μ_Q and μ_P are the means of $\vec{f_Q}$ and $\vec{f_P}$, respectively.

3.1.2 The SSEF Method

The SSEF method uses a triplet of secondary structure elements (SSEs) as a structural fragment. The secondary structure assignment is computed by the DSSP program

[75] and each secondary structure element is approximated by a positional vector in 3D or *an SSE vector*.

Since the relative orientation of distant pairs of secondary structure elements is less stable, we restrict our consideration to triplets that are close in space, requiring each of the three pairwise distances between the midpoints of SSE vectors to be less than a certain threshold. The adoption of "local" SSE triplets as a structural fragment also reduces the effect of an occasional SSE insertion/deletion on footprints of related domains. For example, consider a pair of related domains, one having n SSEs and the other having $n+1$ SSEs. Without any restrictions the additional SSE may generate up to n^2 SSE triplets that will register in the footprint of one structure but not the other. By considering only local SSE triplets the impact of such insertions/deletions is considerably reduced. The particular value of $30\mathring{A}$ that we have adopted reflects a trade-off between noise and the ability to map every structure to an SSE footprint. Smaller threshold values result in a large number of structures with three or more SSEs but no valid SSE triplets. Larger threshold values result in a worse performance as the spatial orientation of triplets becomes less stable and the effect of SSE insertion/deletion grows.

The spatial conformation of an SSE triplet is represented by all pairwise angles and all pairwise distances between the midpoints of the corresponding SSE vectors. Since angles and distances are measured in different units, a standard normalization procedure is applied, normalizing a quantity x by $\frac{x-\text{mean}_x}{\text{stdev}_x}$. The mean and the standard deviation are computed from the distribution of angle and distance values in triplets of the SSE vectors corresponding to structural fragments extracted from the SCOP fold dataset. Given a pair of structural fragments, their distance is then measured by the Euclidean norm of the difference between corresponding points.

The SSEs are either α-helices or β-strands, so in addition to the positional information given by a triplet of vectors in 3D, each structural fragment is assigned a type: $\alpha\alpha\alpha$, $\alpha\alpha\beta$, $\alpha\beta\alpha$, $\alpha\beta\beta$, $\beta\alpha\alpha$, $\beta\alpha\beta$, $\beta\beta\alpha$ or $\beta\beta\beta$, according to the type of secondary structure elements that it contains. From the point of view of protein structure a triplet of α-helices is quite different from a triplet of β-strands even if their spatial conformation is similar. Therefore, a structural footprint generated by the SSEF method is a concatenation of eight structural footprints, one for each SSE triplet type.

The SSEF method selects $1,500$ models to provide an adequate representation of SSE triplets present in the protein structure universe. Since triplets of secondary structure elements with a majority of β-strands are more abundant than other triplets, the method allocates 225 models each to footprints for $\alpha\beta\beta$, $\beta\alpha\beta$, $\beta\beta\alpha$, and $\beta\beta\beta$ triplet types and 150 models each to footprints for $\alpha\alpha\alpha$, $\alpha\alpha\beta$, $\alpha\beta\alpha$, and $\beta\alpha\alpha$ triplet types.

3.1.3 The SEGF Method

The SEGF method uses a contiguous segment (thirty-two residues long) of protein backbone as a structural fragment. The protein backbone is viewed as a polygonal line passing through the C_α atoms whose conformation is captured by a set of fourteen shape descriptors, a subset of the thirty shape descriptors originally used by Rogen *et al.* [104, 103]. The shape descriptors are various combinations of an *average crossing number*, a geometric invariant that captures the relative orientation of two oriented line segments. In what follows we first describe the average crossing number invariant and then show how the fourteen shape descriptors are constructed using this invariant as a building block.

Given an oriented line segment u, we will denote by u^{sp} the coordinates of u's start point and by u^{ep} the coordinates of u's end point. When a pair of segments, u and v, is

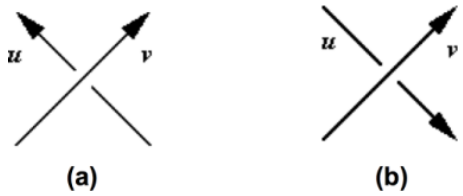


Figure 3.1: When projection of a pair of oriented line segments results in an overcrossing, its value is determined by the right-hand rule involving the projection direction and directions of projected line segments. Here the projection direction is from the page to the reader. **(a)** The value of this overcrossing is $+1$ because the bottom line segment (u) is in the counterclockwise direction from the upper line segment (v). **(b)** The value of this overcrossing is -1 because the bottom line segment (u) is in the clockwise direction from the upper line segment (v).

projected on a plane it produces either zero or one overcrossing. When one overcrossing is produced, it is assigned a value of $+1$ or -1 as shown in Figure 3.1. Thus with every projection direction we can associate a value of either $+1$, -1, or 0 (no overcrossing). The average crossing number between two oriented line segments is the above value averaged over all possible projection directions, projection directions being points on the unit sphere S_2. The value of an overcrossing, $+1$ or -1, is the same for all projection directions that result in an overcrossing. Moreover, projection directions that result in one overcrossing are exactly those that are parallel to vectors of the form $t_v - t_u$, where t_u is a point on u and t_v is a point on v. The above two facts allow us to express the average crossing number as the signed area of a certain parallelogram projected on S_2 and normalized by half of the area of S_2 (half since there is an equivalent parallelogram of directions that correspond to vectors parallel to vectors of the form $t_u - t_v$) as shown in Figure 3.2. The sign of the average crossing number is equal to the sign of $(v^{sp} - u^{sp})^T(v \times u)$. Note that the range of this invariant is the closed interval $[-1, 1]$.

Let us denote by $Wr(u, v)$ the average crossing number between two oriented line segments u and v. Given a polygonal line consisting of r (in our case $r = 31$) oriented line

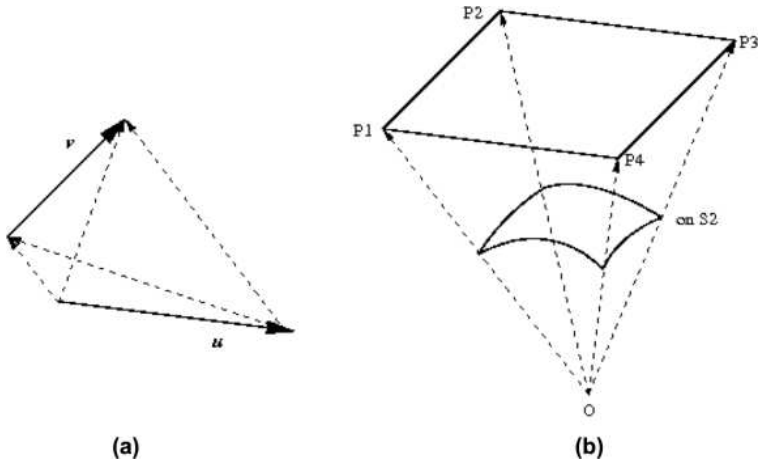


Figure 3.2: **(a)** Projection directions that result in one overcrossing are parallel to vectors of the form $t_v - t_u$, where t_u is on u and t_v on v. **(b)** Those directions trace a parallelogram $P = P_1P_2P_3P_4$, where $P_1 = v^{sp} - u^{sp}$, $P_2 = v^{ep} - u^{sp}$, $P_3 = v^{sp} - u^{ep}$ and $P_4 = v^{ep} - u^{ep}$. The average crossing number equals the signed area of P projected on S_2 and normalized by half of the area of S_2, which can be computed using tools of Spherical Geometry [37].

segments $\{u_1, ..., u_r\}$ the fourteen shape descriptors are constructed in the following way:

$$
\begin{aligned}
I_{(1,2)} &= \sum_{0\le i1<i2\le r} Wr(u_{i1}, u_{i2}) \\
I_{|1,2|} &= \sum_{0\le i1<i2\le r} |Wr(u_{i1}, u_{i2})| \\
I_{(1,2)(3,4)} &= \sum_{0\le i1<i2<i3<i4\le r} Wr(u_{i1}, u_{i2})\ Wr(u_{i3}, u_{i4}) \\
I_{|1,2|(3,4)} &= \sum_{0\le i1<i2<i3<i4\le r} |Wr(u_{i1}, u_{i2})|\ Wr(u_{i3}, u_{i4}) \\
I_{(1,2)|3,4|} &= \sum_{0\le i1<i2<i3<i4\le r} Wr(u_{i1}, u_{i2})\ |Wr(u_{i3}, u_{i4})| \\
I_{|1,2||3,4|} &= \sum_{0\le i1<i2<i3<i4\le r} |Wr(u_{i1}, u_{i2})|\ |Wr(u_{i3}, u_{i4})| \\
I_{(1,3)(2,4)} &= \sum_{0\le i1<i2<i3<i4\le r} Wr(u_{i1}, u_{i3})\ Wr(u_{i2}, u_{i4}) \\
I_{|1,3|(2,4)} &= \sum_{0\le i1<i2<i3<i4\le r} |Wr(u_{i1}, u_{i3})|\ Wr(u_{i2}, u_{i4}) \\
I_{(1,3)|2,4|} &= \sum_{0\le i1<i2<i3<i4\le r} Wr(u_{i1}, u_{i3})\ |Wr(u_{i2}, u_{i4})| \\
I_{|1,3||2,4|} &= \sum_{0\le i1<i2<i3<i4\le r} |Wr(u_{i1}, u_{i3})|\ |Wr(u_{i2}, u_{i4})| \\
I_{(1,4)(2,3)} &= \sum_{0\le i1<i2<i3<i4\le r} Wr(u_{i1}, u_{i4})\ Wr(u_{i2}, u_{i3})
\end{aligned}
$$

$$
\begin{aligned}
I_{|1,4|(2,3)} &= \sum_{0 \le i1 < i2 < i3 < i4 \le r} |Wr(u_{i1}, u_{i4})|\ Wr(u_{i2}, u_{i3}) \\
I_{(1,4)|2,3|} &= \sum_{0 \le i1 < i2 < i3 < i4 \le r} Wr(u_{i1}, u_{i4})\ |Wr(u_{i2}, u_{i3})| \\
I_{|1,4||2,3|} &= \sum_{0 \le i1 < i2 < i3 < i4 \le r} |Wr(u_{i1}, u_{i4})|\ |Wr(u_{i2}, u_{i3})|
\end{aligned}
$$

The distance between a pair of structural fragments is given by Euclidean distance between their representations. The SEGF method selects 300 models to provide an adequate representation of structural fragments in the protein structure universe.

3.2 Experimental Results

3.2.1 Evaluation Procedures

Protein structure comparison methods are commonly benchmarked against two major protein structure classification databases, the CATH [96] and SCOP [91] databases. In this thesis we adopt the CATH classification database as a gold standard for the definition of structurally similar protein domains or *true relationships*. We use the CATH Homologous Superfamily (superfamily) level to measure the method's ability to detect structural similarity between closely related protein domains or *homologs* and the CATH Topology (fold) level to measure the method's ability to detect structural similarity between distantly related protein domains or *topologs*.

In each performance evaluation experiment a protein structure comparison method is used to compare a set of query protein domains to a large database of structures. To focus evaluation on cases where protein sequence comparison methods fail, and thus protein structure comparison methods are of greatest practical value, the database is a set of protein domains where no two domains have sequence identity greater than 35%.

Normally, queries would represent well-populated superfamilies or folds in the database, but different experiments use different sets of query proteins; these are described in detail in Section 3.2.1.1.

Once a query protein domain is compared to every domain in the database, the results of the comparisons can be used to rank the database domains based on their structural similarity to the query. Ideally, database domains related to the query would appear at the top of the list, followed by unrelated domains. We use a number of well-known techniques, which are described in Section 3.2.1.2, to quantify and visualize how far from the ideal ranking the actual ranking is.

3.2.1.1 Data Sets

In this thesis we used the CATH database (version 2.6 released in April 2005) for benchmarking purposes. To create a set of database domains, we downloaded a list of non-redundant domains filtered at 35% sequence identity from the CATH classification database website. We excluded from the list domains for which a valid footprint could not be produced by one or more methods. This resulted in a dataset with $5,588$ domains. The set of database domains contains members from $1,416$ superfamilies.

To compare the SSEF method with well-established protein sequence and structure alignment methods using the data from the study of Sierk *et al.* [113], we closely followed the query set selection procedure described in that study. In particular, we chose the longest domain from each superfamily with at least six members in the set of database domains. There are 196 such superfamilies which resulted in a set of 196 query domains.

For other experiments we first identified a set of well-populated superfamilies. A superfamily is *well-populated* is it satisfies the following constraints: (i) the superfamily

has at least five members in the set of database domains and (ii) the superfamily is not the only superfamily in its fold. There are 133 superfamilies that satisfy the above constraints. We then set the query set to contain all the members of these well populated superfamilies, which resulted in a set of $2,348$ queries.

3.2.1.2 Measuring Retrieval Accuracy

We used *Coverage versus Error* plots [113] to summarize a method's performance at a given classification level. Given a protein structure comparison method and a database of protein domains, each query protein domain defines a Coverage versus Error curve. The curve is computed by first ordering the database domains by their structural similarity to the query domain. This list is examined from the most similar to the least similar domain; for each false positive result (an unrelated domain) the number of errors is incremented and the *coverage level* (the fraction of related domains retrieved so far) is recorded. The curve shows the coverage obtained at each error level. To obtain one curve per method we either took the median coverage values (as in Figure 3.3) or took the average coverage values, first across different queries in the same classification group and then across different classification groups (as in Figure 3.4).

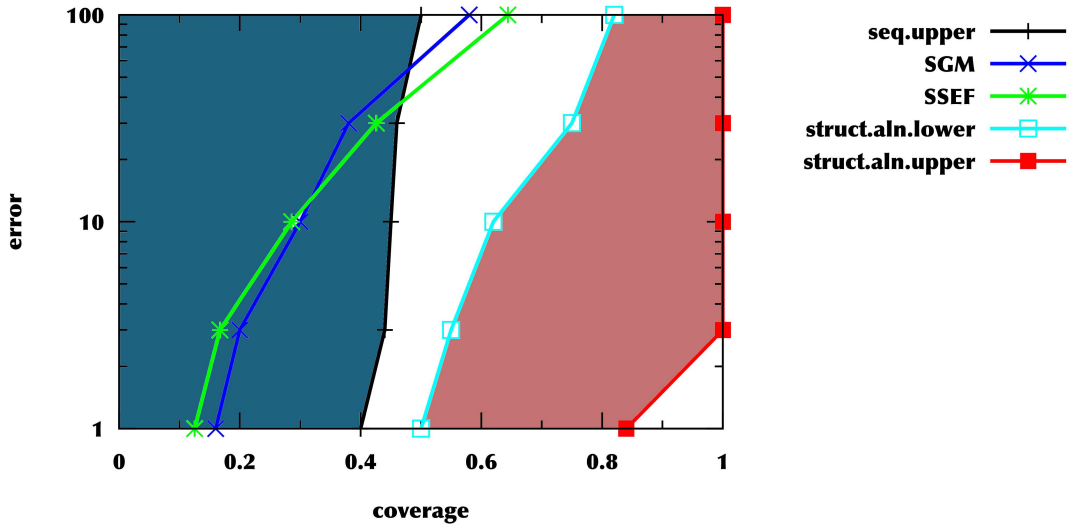
To quantify the method's ability to retrieve other members of a superfamily given one member as a query we used ROC_{300} scores [56]. Briefly, the ROC_n score measures to what extent the related domains precede the unrelated domains among the n highest ranked database domains. We use this measure to compare a methods's performance across individual superfamilies in Figure 3.5; again one value per superfamily was obtained by averaging the scores across different queries in the same superfamily.

3.2.2 Comparison to Common Protein Structure and Sequence Alignment Methods

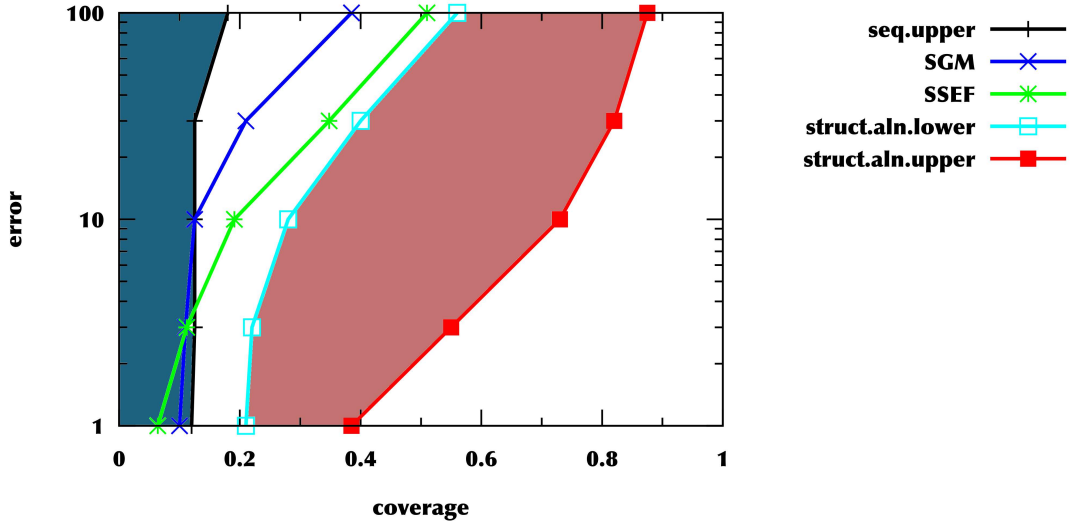
To compare the SSEF method to established sequence and structure alignment methods, we used data from the study of Sierk *et al.* [113], where five full-fledged protein structure alignment methods (DALI [65], STRUCTAL [48], CE [111], VAST [50], and MATRAS [76]), two projection methods (SGM [103] and PRIDE [22]), and two sequence alignment methods (SSEARCH [99] and PSI-PLAST [5]), were evaluated based on their ability to detect relationships at the CATH homologous superfamily and topology levels.

We combined the data for the SSEF method with the data reported in [113] to produce the Coverage versus Error plots in Figure 3.3. (Since the performance of PRIDE was worse than that of SGM, Sierk *et al.* did not include the data for PRIDE in their paper.) To simplify the plots, the outcomes for the five protein structure alignment methods are combined into two curves showing the worst and the best performance. Similarly, the outcomes for the two sequence alignment methods are combined into one curve showing the best performance. It should be noted that the performance of the SSEF method was evaluated with the newer version, version 2.6, of the CATH classification database, whereas the original study of Sierk *et al.* was performed with version 2.3. As the number of non-redundant structures almost doubled from $2,771$ (used in the original evaluation) to $5,588$ (used in evaluation of the SSEF method), we expect the relative performance of the SSEF method to be better than what is shown on the plots.

At the CATH topology level, shown in Figure 3.3**(b)**, the SSEF method has a better coverage than the SGM and sequence alignment methods at all error levels (except error=1). At the CATH homologous superfamily level, as shown in Figure 3.3**(a)**, the coverage achieved by our method at low error levels is significantly worse than that of sequence alignment methods, which are extremely accurate in identifying close homologs.



(a)



(b)

Figure 3.3: Coverage versus Error plots for the SSEF method, SGM, the lower and the upper bounds on the five protein structure alignment methods (struct. aln. lower and struct. aln. upper) and the upper bound on the two sequence comparison methods (seq. upper). The shaded areas highlight the performance boundaries for sequence (blue) and structure (pink) comparison methods. The coverage displayed is the median coverage among the selected queries. Thus, for example in **(a)**, the best structural alignment method retrieves at least 83% of true positive pairs for half of the queries, when the first false positive pair is encountered. **(a)** Pairs in the same CATH homologous superfamily group are true positives; pairs in different CATH homologous superfamily groups are false positives. **(b)** Pairs in the same CATH topology group are true positive; pairs in different CATH topology groups are false positives.

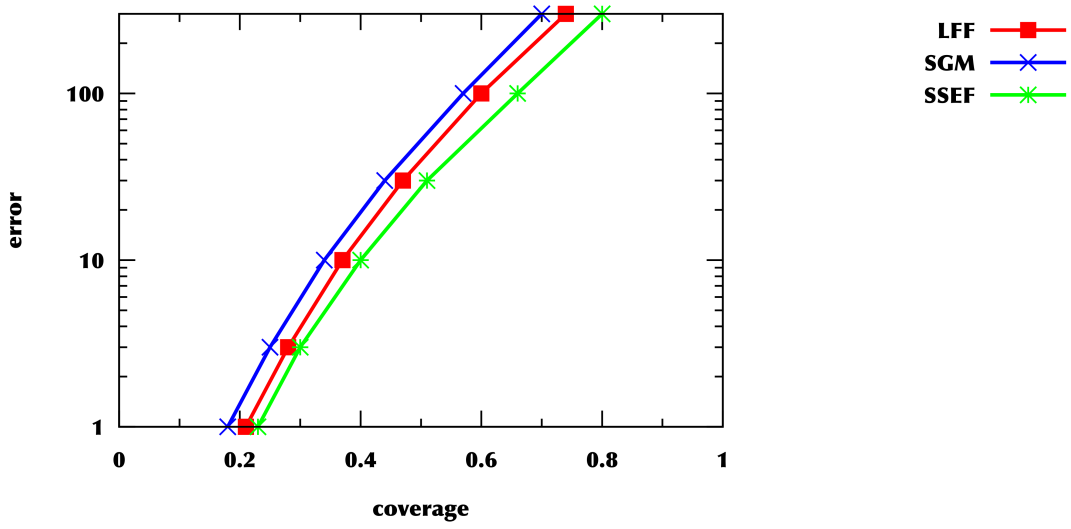
Moreover, while all three projection methods (PRIDE, SGM, and SSEF) are far from achieving the performance of the best protein structure alignment method, the SSEF method performs surprisingly well at high error levels at both the CATH homologous superfamily and topology levels. In particular, at the CATH topology level and error ≥ 30, our method has a comparable performance to that of some of the full-fledged protein structure alignment methods.

3.2.3 Comparison to Other Projection Methods

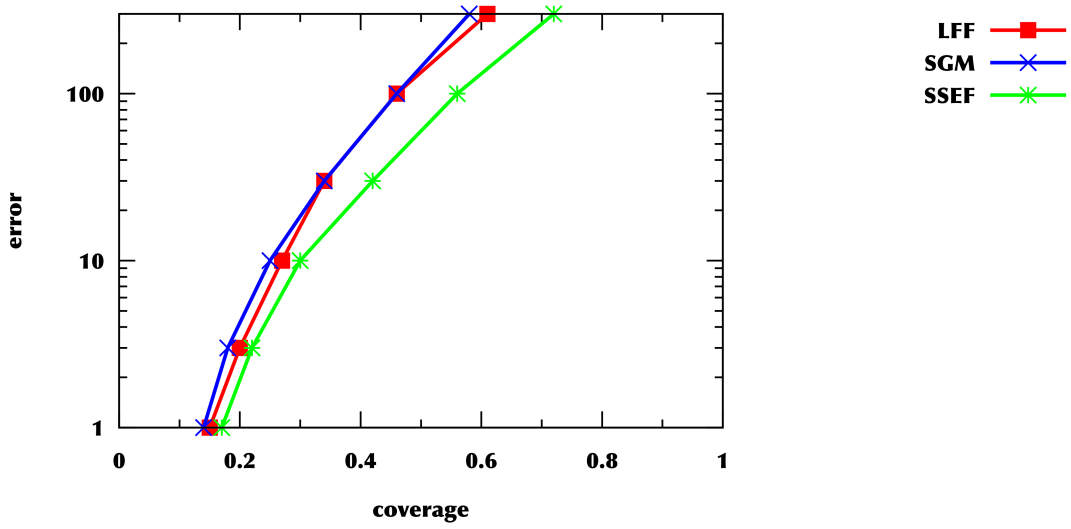
We use Coverage versus Error plots to compare the methods' ability to detect relationships at the CATH superfamily and fold levels. The plots are shown in Figures 3.4(a)-(b). As expected, structural similarity between distantly related domains is more difficult to detect than structural similarity between close homologs for all four methods. Thus, at the superfamily level, the three best methods achieve $70\% - 80\%$ coverage at the 300th false positive and only $58\% - 72\%$ at the fold level. While the SSEF method has better performance at all classification levels, the difference is most profound at the fold level.

To compare the efficiency of the projection methods evaluated in this study we analyze for each method the running time needed to perform all-against-all structure comparison of $5,345$ domains in the SCOP 40%-id dataset. All programs were run on a Linux machine with an Intel Xeon CPU 3.20GHz. The results are shown in Table 3.1.

For any projection method, the all-against-all structure comparison involves two steps. The first step is the pre-processing step where the structures are projected into vectors, and the second step is the pairwise distance computation between the set of vectors. If there are n structures in the dataset then the total running time is $n \times prep + \frac{n(n-1)}{2} \times eval$, where $prep$ is the average pre-processing time per structure and $eval$ is



(a)



(b)

Figure 3.4: Coverage versus Error plots for the SSEF, LFF, and SGM. Data for the PRIDE2 method is not shown as the program supplied by the authors was crashing on the CATH dataset. The performance of the PRIDE2 method on the SCOP dataset, not shown here, is worse than that of the other three methods. **(a)** Pairs in the same CATH superfamily group are true positives; pairs in different CATH homologous superfamily groups are false positives. **(b)** Pairs in the same CATH fold group are true positive; pairs in different CATH fold groups are false positives.

	running time in seconds			
	ssef	lff	sgm	pride
pre-processing	3,067	490,449	4,397	not available
distance computations	1,054	169	136	not available
total	4,121	490,618	4,533	13,200

Table 3.1: The running time (in seconds) to perform all pairwise comparisons of 5,345 domains for the SSEF, LFF, SGM, and PRIDE2 methods. The running time is broken into running times spent on the pre-processing step and the distance computation step. The pre-processing step includes all the computation necessary to compute projections for 5,345 domains. The distance computation step includes all pairwise distance computations between 5,345 projections computed in the pre-processing step. As the detailed information is not available for the PRIDE2 method, only the total time is shown for this method.

the average time to compare a pair of structures. It should be noted that we use the pre-processing to denote the mapping of each structure into a vector; i.e., no pairwise computations are done during this step.

For applications of screening and classifications, we can assume that the pre-processing step is done once for the database proteins and therefore the running time spent on in this step is amortized as the number of queries against the database grows.

The running time spent on distance computations is mainly affected by the dimension of the projection, which we denote by p. Our method uses $p = 1,500$ and takes about 10 times longer to compute the distances than the LFF ($p = 100$) and SGM ($p = 30$) methods. But even the 1,054 seconds to perform $5,345 * (5,345 - 1)/2 = 14,281,840$ protein structure comparisons is almost negligible compared to the time it would take DALI [65] to perform the same number of comparisons. We have used the DaliLite program [67] and estimated that one query against the same database of 5,345 domains takes on average 4,800 seconds or 1.3 hours. Therefore, unless a screening method is applied,the entire all-against-all comparison would take about 3,474 hours or nearly five months to compute.

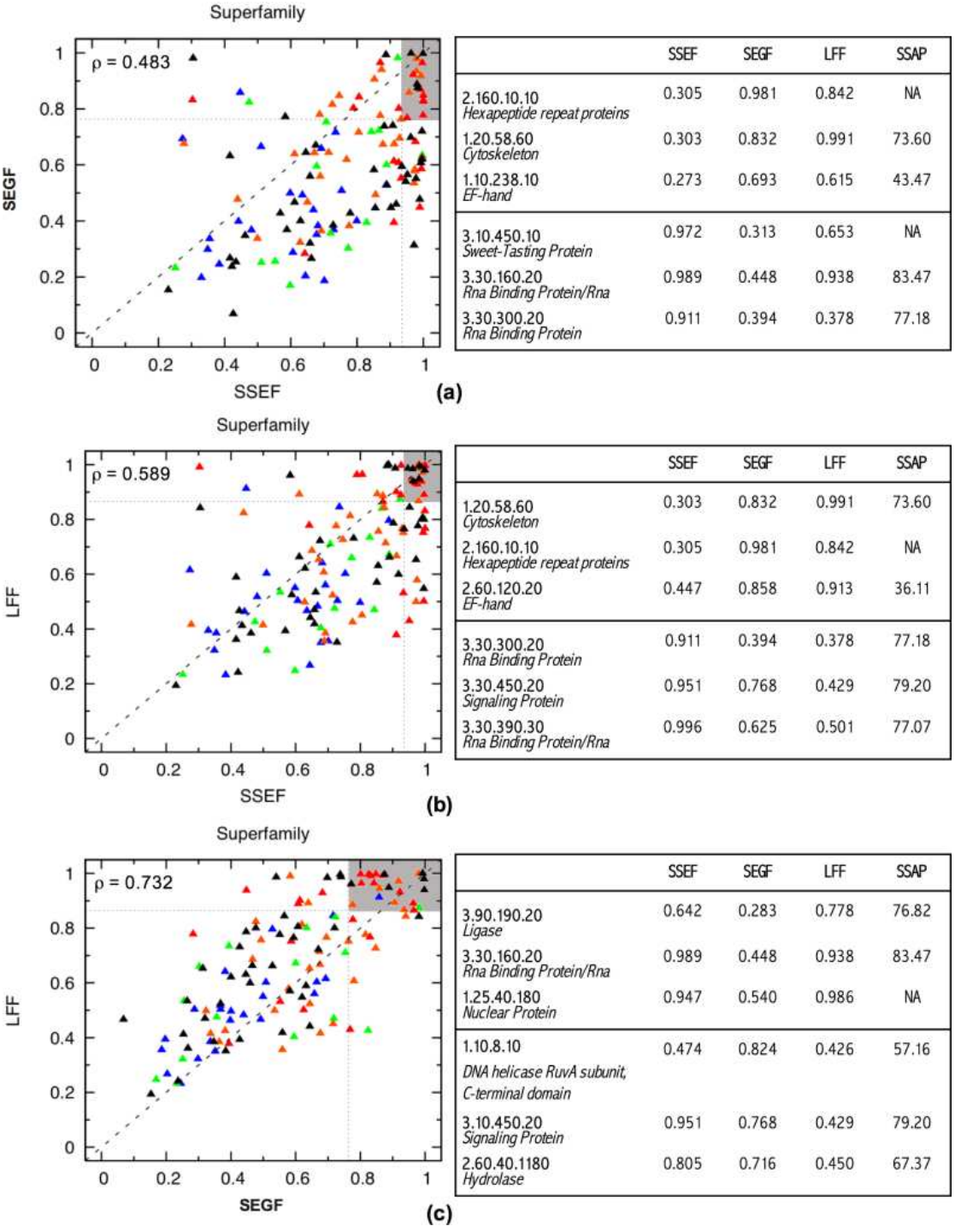
3.2.4 The Impact of the Structural Fragments on Performance

3.2.4.1 Detecting Structural Similarity at the CATH Homologous Superfamily Level

Even though the SSEF method has the best performance on average (see Table 3.3), no method performs consistently the best over all superfamilies. Figure 3.5 shows the ROC_{300} scores as a scatter plot; there is one plot per pair of methods; each superfamily is a point on the plot with the coordinates being the ROC_{300} scores of the corresponding methods. The performance of the methods is poorly correlated, especially that of the SSEF and SEGF methods. The poor correlation can be attributed to the fact that the methods capture different aspects of protein structure in their footprints. Thus structural differences between the members of a superfamily may "confuse" some methods more than others and the amount of confusion depends on how these structural differences affect the structural footprint produced by the method.

To illustrate this point, let us consider two outliers in Figure 3.5, superfamilies for which the performance of one method is quite different from that of another, the `1.20.58.60` (*Cytoskeleton*), and `3.30.300.20` (*Rna Binding Protein*) superfamilies.

The poor performance of the SSEF method on the `1.20.58.60` superfamily can be partially attributed to variability in secondary structure assignment as shown in Figure 3.6(a). Since the second helix in the `1quuA2` domain is split into two helices, the `1quuA2` domain has four SSE triplets that participate in the footprint construction, while the `1cunA1` domain has only one such triplet. In this case, the structural change that produced an additional SSE is small and therefore both the SEGF and LFF methods perform well since they do not use secondary structure information. In general, the SSEF method is most sensitive to structural changes that affect the number and/or relative orientation

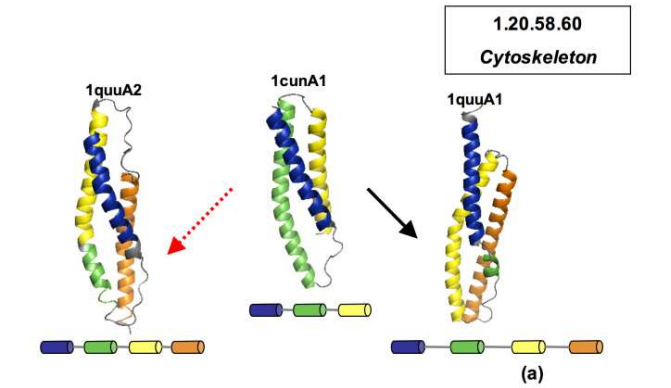


	SSEF	SEGF	LFF	SSAP
2.160.10.10 *Hexapeptide repeat proteins*	0.305	0.981	0.842	NA
1.20.58.60 *Cytoskeleton*	0.303	0.832	0.991	73.60
1.10.238.10 *EF-hand*	0.273	0.693	0.615	43.47
3.10.450.10 *Sweet-Tasting Protein*	0.972	0.313	0.653	NA
3.30.160.20 *Rna Binding Protein/Rna*	0.989	0.448	0.938	83.47
3.30.300.20 *Rna Binding Protein*	0.911	0.394	0.378	77.18

	SSEF	SEGF	LFF	SSAP
1.20.58.60 *Cytoskeleton*	0.303	0.832	0.991	73.60
2.160.10.10 *Hexapeptide repeat proteins*	0.305	0.981	0.842	NA
2.60.120.20 *EF-hand*	0.447	0.858	0.913	36.11
3.30.300.20 *Rna Binding Protein*	0.911	0.394	0.378	77.18
3.30.450.20 *Signaling Protein*	0.951	0.768	0.429	79.20
3.30.390.30 *Rna Binding Protein/Rna*	0.996	0.625	0.501	77.07

	SSEF	SEGF	LFF	SSAP
3.90.190.20 *Ligase*	0.642	0.283	0.778	76.82
3.30.160.20 *Rna Binding Protein/Rna*	0.989	0.448	0.938	83.47
1.25.40.180 *Nuclear Protein*	0.947	0.540	0.986	NA
1.10.8.10 *DNA helicase RuvA subunit, C-terminal domain*	0.474	0.824	0.426	57.16
3.10.450.20 *Signaling Protein*	0.951	0.768	0.429	79.20
2.60.40.1180 *Hydrolase*	0.805	0.716	0.450	67.37

Figure 3.5: There is one scatter plot per pair of methods: SSEF and SEGF **(a)**, SSEF and LFF **(b)**, and SEGF and LFF **(c)**. Each superfamily is a point on the plot with the coordinates being the ROC_{300} scores of the corresponding methods. For every pair of methods, six superfamilies that deviate the most from the diagonal are listed in the table adjacent to the plot. The superfamilies are colored according to the minimum SSAP score for a pair of domains in the superfamily as reported by the DHS database [21]: blue for scores in $(0.0, 53.44]$, green for scores in $(53.44, 63.32]$, orange for scores in $(63.32, 73.48]$, and red for scores in $(73.48, 100.00]$. The SSAP score measures the structural similarity on a scale from 100.0 (the most similar) to 0.0 (the least similar). Our chosen threshold values, 53.44, 63.32, and 73.48, correspond to the 25th, 50th, and 75th percentile respectively. The superfamilies for which the SSAP scores are not available are colored black. The correlation between the performance of every pair of methods is captured by Pearson correlation coefficient which is shown in upper left corner of the corresponding plots.



related domains	SSEF	SEGF	LFF
1cunA2	1	2	0
1hciA4	27	6	0
1quuA1	82	35	1
1hciA1	390	66	4
1g8xB7	452	66	4
1g8xA6	1154	77	4
1quuA2	1236	88	12

related domains	SSEF	SEGF	LFF
1hh2P2	0	28	1374
1k0rB2	0	222	2640
1egaA2	0	68	9
1josA0	0	7	239
1lqlA2	0	494	189
1n2fA2	0	1223	654
1k0rA1	0	6	1462
1ml8A2	2	651	102
1qwiA0	77	639	271

Figure 3.6: **(a)** The `1.20.58.60` (*Cytoskeleton*) superfamily. In the table to the right, for each database domain related to the query `1cunA1`, we show the number of errors encountered before the domain is retrieved. Both the SEGF and LFF methods retrieve all seven related domains before the 300th error. (In this case, any domain in a fold group other than `1.20.58` is counted as an error.) In contrast, the SSEF method retrieves only `1cunA2`, `1hciA4`, and `1quuA1`. The structure of the query domain `1cunA1` and two related domains are shown on the left, colored according to secondary structure assignments and also schematically represented by diagrams adjacent to the structures. The secondary structure assignment was computed using the DSSP (Dictionary of Protein Secondary Structure) program [75]. **(b)** The `3.30.300.20` (*Rna Binding Protein*) superfamily. Given the `1fjgC1` Domain as a query, the SSEF method retrieves all nine related domains before the 300th error. On the other hand, the SEGF and LFF method retrieve only five related domains. The ranking of the related domains is summarized in the table to the right. The structure of the query domain `1fjgC1` and two related domains are shown on the left, colored according to secondary structure assignments and also schematically represented by diagrams adjacent to the structures. The protein structures were rendered using PyMOL [30].

of SSEs.

In contrast, the structural variability exhibited by the members of the 3.30.300.20 superfamily does not affect the performance of the SSEF method, but it does affect the other two methods. As shown in Figure 3.6)(b), the members of this superfamily have approximately the same number of SSEs, and they are oriented in roughly the same way.

It is reasonable to assume that for structurally conserved superfamilies all three methods would perform well. To check this hypothesis we color coded the points in the scatter plots of Figure 3.5 according to the structural diversity of the corresponding superfamilies, where red denotes the most structurally conserved superfamilies and blue the least structurally conserved superfamilies. Even though the concentration of the red points in the upper-right corner is clearly visible on all three plots, there are structurally conserved superfamilies for which one or more methods do not perform well. This can happen when a small structural change triggers a big change in the structural footprint produced by the method; consider for example performance of the SSEF method on the 1.20.58.60 superfamily discussed above. Another reason for poor performance of a method on a structurally conserved superfamily is its inability to distinguish between the members of the superfamily and members of other superfamilies that are composed of similar structural fragments but have different overall structure.

3.2.4.2 Combining Structural Footprinting Methods

Can we take advantage of variation in performance of the methods across different superfamilies, i.e., can the output of the methods be combined in such a way as to leverage their relative strengths? To answer this question we have studied two combination strategies: voting and linear combination of similarity scores.

	w_{SSEF}	w_{SEGF}	w_{LFF}	w_0
SSEF+SEGF+LFF	6.08	3.16	2.85	8.57
SSEF+SEGF	7.28	3.72	N/A	7.26
SSEF+LFF	7.48	N/A	4.18	8.56
SEGF+LFF	N/A	6.34	6.38	9.76

Table 3.2: Coefficient values learned with SVM for the four combinations: SSEF+SEGF+LFF, SSEF+SEGF, SSEF+LFF, and SEGF+LFF.

In voting, each method's similarity scores are first used to rank the database domains. The new score of a database domain is determined by averaging the domain's positions in the three original rankings, with ties being resolved arbitrarily.

In linear combination, a new structural similarity score between the query and a database domain is defined as a linear combination of the original similarity scores:

$$sim_{\text{COMB}} = w_{\text{SSEF}}\ sim_{\text{SSEF}} + w_{\text{SEGF}}\ sim_{\text{SEGF}} + w_{\text{LFF}}\ sim_{\text{LFF}} - w_0.$$

The coefficients (w_{SSEF} , w_{SEGF} , and w_{LFF}) are learned using the Support Vector Machine (SVM) learning algorithm [28] from a set of positive and negative examples. For each well-populated superfamily we selected uniformly at random 10 pairs of domains where both domains are from the superfamily to form the set of positive examples, and 10 pairs of domains where one domain is from the superfamily and another domain is from a different fold to form the set of negative examples. Therefore each set contains 1330 domain pairs, 10 pairs for each of the 133 well-populated superfamilies. We used the SVMLight implementation [74] of the SVM learning algorithm with default parameters. The set of coefficients learned is summarized in Table 3.2.

As shown in Table 3.3, the average ROC_{300} scores increase from 0.750 (the SSEF method), to 0.774 (using the voting combination strategy), and to 0.814 (using the linear combination strategy). Even with the simple voting strategy we obtain an improvement of 0.024 over the best (on average) method; the introduction of weights (in linear combination

SSEF	SEGF	LFF	voting
0.750	0.581	0.665	0.774

linear combination			
SSEF+SEGF+LFF	SSEF+SEGF	SEGF+LFF	SEGF+LFF
0.814	0.798	0.789	0.677

Table 3.3: The average ROC_{300} scores obtained over a range of combination strategies: the original methods, voting with all three methods, and linear combination of similarity scores.

strategy) further improves the performance by 0.040. We used the *binomial sign test for two dependent samples* [110] to evaluate the statistical significance of improvements due to combination. This test can be applied to evaluate whether a number of superfamilies on which one method outperforms the other differs significantly from what would be expected by chance. We found that both combination strategies significantly improve over the SSEF method: the improvement due to voting has a p-value of $3.35e$-02 and improvement due to linear combination has a p-value of $1.43e$-15.

The success of a combination strategy largely depends on how consistent are the methods in their ranking of false positives. The combination is most effective when the methods disagree on their ranking of false positive domains, i.e., false positive domains ranked near the top by one method are ranked near the bottom by other methods. Thus the success of a combination strategy is a function of the methods being combined. To find out which pair of methods are the most complementary, i.e., their combination gives the best results, we repeated the linear combination experiments for all pairs of methods. The outcomes of these experiments (see Table 3.3 under SSEF+SEGF, SSEF+LFF, and SEGF+LFF) indicate that combination of the SSEF and SEGF methods gives the best results. This outcome demonstrates that the stand-alone performance is of lesser importance for combination purposes. Indeed, while the SEGF method is the weakest among

the three methods, its performance is the least correlated with that of the SSEF method (see the performance correlation values in Figure 3.5).

3.3 Summary

Projection methods are a class of fast protein structure comparison methods that achieve a considerable speed-up over full-fledged protein structure alignment methods by mapping a protein structure to a high-dimensional vector. Once the mapping is done the structural similarity is approximated by a distance computation between the corresponding vectors. In the process of mapping some structural information is lost. Thus, the central issue in designing a good projection method is how to define a mapping that is able to capture all the salient features of protein structure.

In this thesis we systematically addressed this issue by introducing the structural footprinting framework. Our framework defines a family of projection methods that differ in the "structural alphabet" used by the method to describe protein structure. In fact, a large variety of methods can be generated that emphasize different aspects of protein structure.

We demonstrated that structural footprinting is a useful approach for designing fast protein structure comparison methods. We also explored how the retrieval accuracy of a structural footprinting method depends on the structural alphabet used. We found that a method whose structural alphabet incorporates secondary structure information and completely ignores less conserved loop regions has the best performance on average in retrieving evolutionarily related protein pairs. We also found that combining structural footprinting methods that use complementary structural alphabets significantly improves performance and allows the combined method to better tolerate various types of structural

variability exhibited by groups of evolutionarily related proteins.

Chapter 4

Protein-Protein Interactions Preliminaries

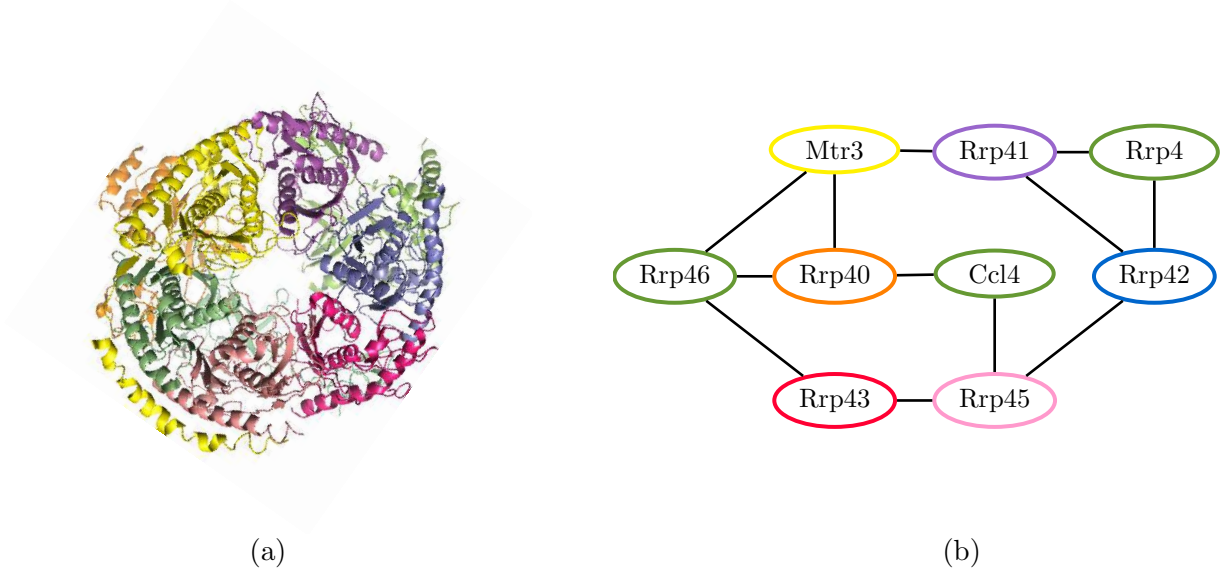
The purpose of this chapter is to provide the reader with relevant background information on protein interactions and protein interaction networks. Over the years several experimental techniques were developed that allow the inference of protein interactions. These techniques are reviewed in Section 4.1 along with databases where currently known protein interactions are deposited.

Protein interactions are usually represented by a graph, a *protein interaction network*, where nodes are proteins and edges are interactions between the proteins. Due to high error rates in experimentally determined protein interactions [90, 119, 29, 115], the transition from experimental data to a reliable protein interaction network is not straightforward. Protein interaction networks used in this thesis and the computational approaches used to derive them are described in Section 4.2.

4.1 Experimental Techniques for Determining Protein Interactions

Two currently known experimental techniques that can be used to determine protein interactions on a large scale are *yeast two-hybrid* (Y2H) [38] and *complex purification* [102].

The yeast two-hybrid is targeted at detecting *physical* or *binary protein interactions*. A pair of proteins physically interact if they directly bind each other. Figure 4.1(a) shows the structure of the *human exosome complex*, the molecular machine responsible for RNA destruction. This complex has nine protein subunits, but not every pair of subunits come



(a) (b)

Figure 4.1: **(a)** A schematic representation of the crystal structure of the human exosome complex (PDB code `2nn6`). The complex contains nine protein subunits shown in different colors. **(b)** Physical interactions between the subunits of the complex are schematically represented by a graph, where there is an edge between a pair of subunits if and only if they physically interact.

into close contact with each other; for example, there no physical interaction between the `Rrp45` and `Rrp40` subunits. Physical interactions among the subunits of the exosome complex are shown in Figure 4.1(b).

The original yeast two-hybrid technique takes advantage of the `GAL` transcription activator in *Saccharomyces cerevisiae*, which is required for expression of genes encoding enzymes of galactose utilization. The `GAL4` protein has two functionally essential domains: the *binding domain* is responsible for binding the promoter sequence and the *activation domain* is required for transcription activation. To determine whether proteins X and Y are able to directly bind each other, protein X is fused to the binding domain and protein Y is fused to the activation domain. Physical interaction between X and Y brings the domains of the `GAL4` protein in proximity, which results in transcription of the regulated genes. The expression level of these genes is monitored and serves as a measure of physical interaction between proteins X and Y. The yeast two-hybrid technique is applied in small-

scale experiments to detect physical interactions between specific pairs of proteins [81]. Several recent studies have applied the technique on a genome-wide scale to map protein interactions in *Saccharomyces cerevisiae* [118, 70].

The complex purification technique, on the other hand, is targeted at detecting components of multi-protein complexes. To characterize proteins that are in the same complex/complexes with a *bait protein*, the bait coding gene is tagged (fused) with a DNA sequence which permits easy purification. The tag is used later on to pull out the bait protein together with all its associated proteins or *preys* using techniques such as co-immunoprecipitation or tandem affinity purification. The preys are subsequently identified by mass spectrometry. The complex purification technique was applied on a genome-wide scale to characterize protein complexes in *Saccharomyces cerevisiae* [64, 45, 80, 46].

There are numerous databases that store experimentally determined protein interactions [88, 107, 77, 116]. For example, as of December 2005, the Database of Interacting Proteins (DIP) [107] catalogs 18,224 interactions among 4,936 proteins in *Saccharomyces cerevisiae*, which were obtained in 22,340 experiments.

4.2 Protein Interaction Networks

Several independent assessments of protein interaction data derived from high-throughput experiments found that these interactions contain a large number of false positives [90, 119, 29, 115]. For example, Deane *et al.* estimated that interactions reported in several genome-wide yeast-two-hybrid screens in *Saccharomyces cerevisiae* [40, 41, 118, 69, 70] contain as much as 50% false positive interactions.

To circumvent high error rates in protein interaction data, computational methods have been proposed to construct reliable protein interaction networks that rely on other

sources of interaction evidence [29, 72, 10, 46, 80, 25]. These methods resulted in several reliable genome-wide protein interaction networks for *Saccharomyces cerevisiae*, five of which are used in Chapter 5 and Chapter 6 of this thesis and therefore are briefly described here.

In addition to estimating the fraction of false positive interactions in the yeast-two-hybrid high-throughput screens, Deane *et al.* proposed a computational method that combines protein interaction data with sequence information to derive a subset of reliable protein interactions [29]. The method builds upon an observation that if two proteins, P_1 and P_2, interact, then so do their *paralogs*, proteins in the same organism having high sequence similarity to P_1 and P_2. Consequently, the method assigns a confidence score to the interaction between P_1 and P_2 based on the number of observed interactions between two protein families, one family being the proteins similar to P_1 and another, proteins similar to P_2. The method is applied on a regular basis to interactions deposited into the DIP database [107] to derive a subset of high-confidence interactions. In later sections we refer to this high-confidence subset of interactions as the *DIP CORE network*.

Jansen *et al.* proposed a computational method that uses Bayesian Networks to predict which pairs of proteins interact [72]. The method trains a Bayesian network that combines a variety of genomic features such as mRNA co-expression, co-localization, etc., to derive interaction confidence scores for protein pairs. The authors used protein interactions derived from a set of manually-curated protein complexes [88] as the set of positive training examples and pairs of proteins localized to different cellular compartments as the set of negative training examples. In later sections we refer to this network as the *BAYESIAN network*.

Recently, Reguly *et al.* [101] manually curated an impressive number of over $31,000$

abstracts and online publications to compile a comprehensive set of protein interactions that were reported in small-scale experiments. This network is believed to consist of biologically relevant protein interactions since interactions reported in small-scale experiments are usually validated by a variety of methods. In later sections we refer to this network as the *LC network* (Literature Curated network).

Since an interaction detected using different experimental techniques is deemed to be reliable, one can filter out potential false positives by intersecting several experimental datasets. This approach was taken by Batada *et al.* [10] who compiled a protein interaction network from protein interactions reported by at least two independent experiments. In later sections we refer to this network as the *HC network* (High Confidence network).

Collins *et al.* [25] derived a protein interaction network from raw purification data reported in two recent genome-wide complex purification experiments [46, 80]. The general idea is to assign to each experimentally identified interaction a confidence score, which takes into account the number of direct (one protein pulls the other) and indirect (both proteins are pulled by a third protein) co-purifications. In later sections we refer to this network as the *TAP-MS network*.

Chapter 5

Dynamic Formation of Multiprotein Complexes [2]

In 1999, Hartwell *et al.* [61] introduced the notion of a *functional module*, a group of cellular components and their interactions that can be attributed a specific cellular function. Some modules are formed from *stable associations*, such as the ribosome, which consists of more than 80 ribosomal proteins and four RNA molecules. Other modules involve *transient associations*, where a protein may associate with different partners at different stages of a cellular process. In addition, there are functional modules that involve a coordinated formation of multi-protein complexes and whose function critically depends on the order in which the interactions occur.

As described in Section 4.1, mature experimental techniques exist that allow the inference of protein interactions, and recent proteomic studies used these and other technologies to characterize protein interactions among the components of many cellular processes [6, 19]. Even though the protein interaction networks for many cellular processes are available, we have little knowledge of the dynamical properties of protein interactions involved in these processes. The main objective of the research effort described in this chapter was directed at bridging this gap by developing a computational approach to extract the dynamical properties of protein interactions from the inherently static topology of a protein interaction network.

[2] This chapter is derived from "Decomposition of overlapping protein complexes: A graph theoretical approach for analyzing static and dynamic protein associations" by E. Zotenko, K. S. Guimaraes, R. Jothi, and T. M. Przytycka, Algorithms for Molecular Biology, 1(1):7, 2006.

Even though protein interaction networks do not explicitly capture the dynamic nature of protein interactions, there are graph theoretic tools that under certain assumptions allow the extraction of this information from the network. In particular, there are graph families whose members have alternative representations that can be used to reason about the dynamics of corresponding protein interactions. Unfortunately, research attempts in this direction are very limited. In fact, we are aware of only one method, due to Farach-Colton *et al.* [34], that uses *interval graphs* to reason about the order in which proteins join the ribosome maturation pathway [35]. *Interval graphs* are a family of graphs whose members have an *interval representation*. An interval representation of a graph is a set of closed intervals on a real line such that there is a one-to-one mapping between the nodes of the graph and the intervals in the set, and a pair of nodes are adjacent if and only if the corresponding intervals intersect.

In their paper, Farach-Colton and colleagues proposed an interval model to represent the assembly pathway of the 60S ribosomal particle. In this model an auxiliary protein "enters" the pathway at some point and "leaves" the pathway at a later point, never to enter the pathway again. The model further assumes that a protein participates in the pathway through binding to other proteins currently in the pathway; therefore the assembly line can be thought of as an evolution of one protein complex to which proteins bind as they enter the pathway and from which proteins dissociate as they leave the pathway. Under this model the protein interaction network that spans the auxiliary proteins involved in the pathway is an interval graph. Indeed, each auxiliary protein corresponds to an interval and two proteins interact if and only if their intervals overlap. Therefore, the protein interaction network can be used to reconstruct the order in which the auxiliary proteins join the pathway.

In this thesis we developed a method, the *Complex Overlap Decomposition* (COD) method, that considerably generalizes the approach taken by Farach-Colton and colleagues. Our method relies heavily on important results from graph theory. More specifically, it uses chordal graphs and their corresponding clique tree representation to model complex formation during a given cellular process and uses cographs and their corresponding modular decomposition to model protein complexes and their variants. The relevant graph theoretic tools for these graph families are described in Section 5.1, and the COD method is described in Section 5.2.

We applied the COD method to two protein interaction networks underlying well studied cellular processes in *Saccharomyces cerevisiae* (bakers yeast): the mating pheromone signaling pathway and the DNA replication module. The description of the cellular processes and their corresponding representations produced by our method are described in Section 5.3.1 and Section 5.3.2 respectively.

5.1 Graph Theoretic Tools

In general, graphs are not required to have any type of regularity. This makes them very flexible combinatorial objects, which are able to represent complex and diverse relationships. In practice, however, graphs that model real world phenomena often have a special structure, which can be revealed through alternative graph representation and/or graph decomposition techniques. Our method builds on two such techniques, a clique tree representation for chordal graphs and modular decomposition for cographs. In this section we briefly review relevant graph-theoretical results for these two graph families. Our review of chordal graphs and their clique tree representation is based on the book by McKee and McMorris [85]; there are several other texts that provide thorough treatment of the

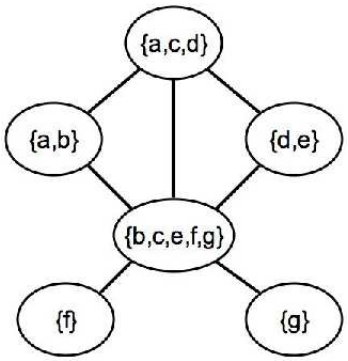


Figure 5.1: The intersection graph of the family of subsets $\mathcal{F} = \{\{a,b\},\{a,c,d\},\{d,e\},\{b,c,e,f,g\},\{f\},\{g\}\}$.

subject, such as the classical treatment by Golumbic [53] and a chapter "An introduction to chordal graphs and clique trees" by Blair and Peyton in [16]. Our treatment of cographs and modular decomposition follows the seminal paper by Corneil *et al.* [26].

We assume that all graphs are undirected and connected. We denote by $G = (V, E)$ an undirected graph with nodes specified by V and edges specified by E, and by $G_S = (S, E_S)$ an *induced subgraph* of G, where $S \subseteq V$ and $E_S = \{(v,w) \in E \mid v,w \in S\}$. For a node $v \in V$, we use $\mathcal{N}(v)$ to denote the set of v's neighbors in G, i.e., $\mathcal{N}(v) = \{u \mid (v,u) \in E\}$. We extend this notation to an arbitrary set of nodes $V' \subset V$ by letting $\mathcal{N}(V') = (\cup_{v \in V'} \mathcal{N}(v)) \setminus V'$.

5.1.1 Chordal Graphs and Clique Tree Representation

Let $\mathcal{F} = \{R_1, ..., R_n\}$ be a family of subsets of some set R. The *intersection graph of* $\mathcal{F}$ is a graph $G = (V, E)$ where $V = \mathcal{F}$ and $E = \{(R_i, R_j) \mid R_i \cap R_j \neq \emptyset\}$; i.e., the nodes of the graph are the subsets in $\mathcal{F}$ and there is an edge between two nodes (subsets) if their intersection is not empty. For example, the intersection graph of $\mathcal{F} = \{\{a,b\},\{a,c,d\},\{d,e\},\{b,c,e,f,g\},\{f\},\{g\}\}$ is shown in Figure 5.1.

It can be shown that any graph is isomorphic to the intersection graph of some

family of subsets; the family of subsets can be thought as an alternative representation of the graph and is called a *set representation* of the graph. A variety of well known graph classes can be characterized by putting restrictions on set representations of graphs in the class. For example, an *interval graph* is isomorphic to the intersection graph of a family of closed intervals on the real line, a *chordal graph* is isomorphic to the intersection graph of a family of subtrees of a tree, and a *disk graph* is isomorphic to the intersection graph of a family of disks on the plane.

In a cycle, a *chord* is any edge that connects two non-consecutive nodes of the cycle. A *chordal graph* is a graph that does not contain chordless cycles of length greater than three. Even though the study of chordal graphs goes back to 1958, the characterization in terms of allowable set representations was given only in 1974 by Gavril [47]. In his seminal paper Gavril established that a graph is chordal if and only if it is isomorphic to the intersection graph of a family of subtrees of a tree; the tree and the family of subtrees are called a *tree representation* of the chordal graph. Figure 5.2(b) shows a tree representation of a chordal graph in Figure 5.2(a).

A *maximal clique* in a graph is a subset of nodes that form a maximal complete induced subgraph. Given a graph $G = (V, E)$, we will use $\mathcal{Q}(G)$ to denote the set of all maximal cliques in G and $K(G)$ to denote the *clique graph* of G. The clique graph of G is the intersection graph of $\mathcal{Q}(G)$, i.e., nodes of $K(G)$ are maximal cliques in G and there is an edge between a pair of nodes (maximal cliques) if their intersection is not empty. Let us illustrate these definitions for the graph $G = (V, E)$ in Figure 5.2(a). This graph has four maximal cliques, which are shown in Figure 5.2(c). The clique graph $K(G)$ is shown in Figure 5.2(d); it has four nodes Q_1, Q_2, Q_3, and Q_4, and is complete since every pair of nodes (maximal cliques) has a non-empty intersection. (In this case all maximal cliques

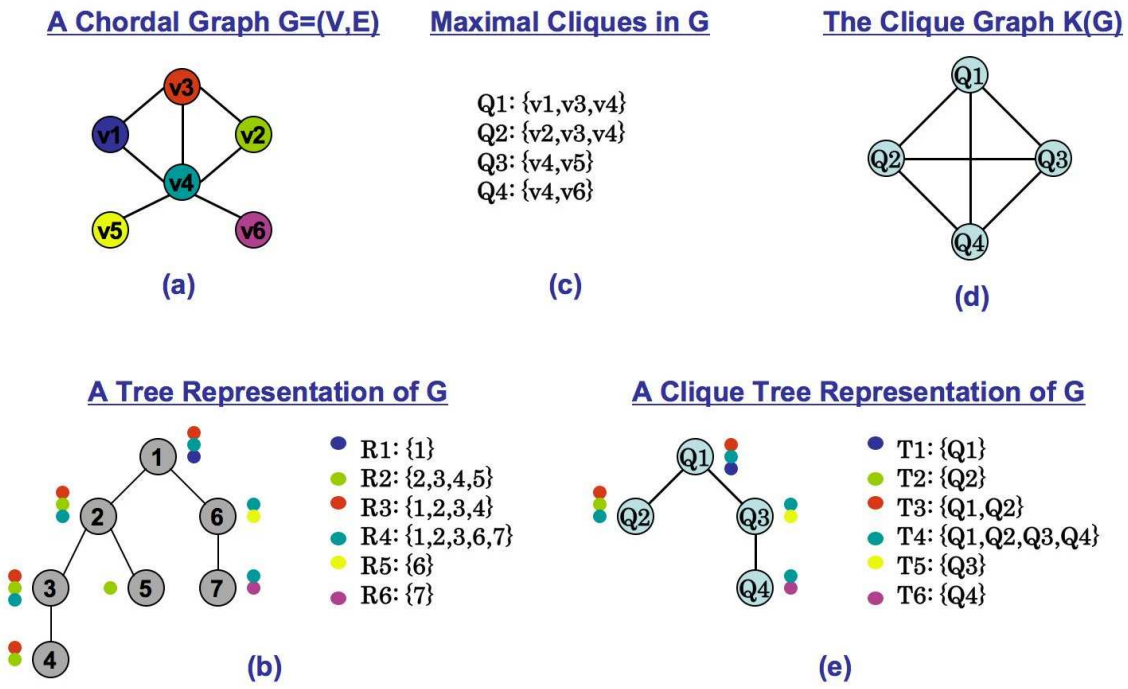


Figure 5.2: **(a)** A chordal graph $G = (V, E)$. **(b)** A tree representation of G: the tree is on the left and the family of subtrees is on the right. Every R_i is schematically shown on the tree by putting a colored circle next to its nodes. For example, R_2 is shown by green circles. **(c)** The set of maximal cliques in G. There are four maximal cliques in the graph, Q_1, Q_2, Q_3, and Q_4. **(d)** The clique graph of G. The clique graph is the intersection graph of $\{Q_1, Q_2, Q_3, Q_4\}$. **(e)** A clique tree representation of G: the clique tree is on the left and the family of subtrees is on the right.

contain node $v_4 \in V$ of the original graph G.)

The above definitions lead us to another result established by Gavril in [47]; every chordal graph $G = (V, E)$ has a special tree representation, the so called *clique tree representation*, in which the tree is a spanning tree of $K(G)$ and the family of subtrees $\mathcal{F} = \{T_1, ..., T_{|V|}\}$ is defined by setting each T_i to the set of maximal cliques that contain a node $v_i \in V$. For example, Figure 5.2(e) shows a clique tree representation for a chordal graph in Figure 5.2(a).

In what follows we show how one can enumerate all possible clique tree representations for a given chordal graph $G = (V, E)$. Let us denote by $\mathcal{Q}(v)$ a set of maximal cliques in the graph that contain a node $v \in V$, i.e., $\mathcal{Q}(v) = \{Q \in \mathcal{Q}(G) \mid v \in Q\}$. In any clique tree representation of G, the family of subtrees is completely determined by the set $\{\mathcal{Q}(v) \mid v \in V\}$, and therefore is unique. What distinguishes one clique tree representation of the graph from another is the spanning tree of $K(G)$ used in the representation; we call this tree a *clique tree*. As every $\mathcal{Q}(v)$ has to be a connected subgraph of a clique tree, the set $\{\mathcal{Q}(v) \mid v \in V\}$ can be thought as a set of constraints that have to be satisfied by any clique tree; i.e., a clique tree is a spanning tree of $K(G)$ for which every $\mathcal{Q}(v)$ is connected. The power of these constraints depends on the structure of the graph, so there are graphs with a unique clique tree and there are graphs for which almost any spanning tree of $K(G)$ is a valid clique tree. It was shown [15] that clique trees are exactly maximum weight spanning trees of $K(G)$, where the weight function on the edges of $K(G)$ is defined as the amount of overlap between two maximal cliques, i.e., $w(Q, Q') = |Q \cap Q'|$. Although the above relation gives an immediate algorithm to enumerate all the clique trees for a chordal graph, we use an approach by Ho and Lee [63] that builds on a connection between edges of a clique tree and the set of minimal vertex separators in the graph, since

it provides a better insight into the source of non-uniqueness.

Given a graph $G = (V, E)$ and a subset of nodes $S \subset V$, S is an *xy-separator* for nodes x and y if the removal of S from the graph disconnects x and y, i.e., x and y are in two different connected components of $G_{V \setminus S}$. We say that S is a *minimal xy-separator* if it is an xy-separator and no proper subset of S disconnects x and y. Finally, S is a *minimal vertex separator* if it is a minimal xy-separator for some x and y in the graph. We denote by $\Delta(G)$ the set of all minimal vertex separators in G. For example, $\Delta(G)$ for the graph in Figure 5.3(a) contains two minimal vertex separators, $S_1 = \{v_3, v_4\}$ and $S_2 = \{v_4\}$. The separator S_1 is a minimal separator for nodes v_1 and v_2 but not for nodes v_5 and v_6 since the removal of $S_2 \subset S_1$ from the graph also disconnects v_5 and v_6.

Let S be a minimal vertex separator in a graph $G = (V, E)$. The removal of S from the graph creates several connected components. Each such connected component C has no neighbors outside of S, i.e., $\mathcal{N}(C) \subseteq S$. A connected component that is adjacent to every element of S ($\mathcal{N}(C) = S$) is a *full connected component* of S. We denote by $\mathcal{C}(S)$ the set of all full connected components of S. For a full connected component $C \in \mathcal{C}(S)$ we denote by $\mathcal{K}(C)$ the set of all maximal cliques in the graph that are contained in $C \cup S$ and contain S, i.e., $\mathcal{K}(C) = \{Q \in \mathcal{Q}(G) \mid Q \subset C \cup S \text{ and } S \subset Q\}$. For example, Figure 5.3(e) shows full connected components for separator $S_2 = \{v_4\}$; in this case $\mathcal{C}(S_2) = \{C_1, C_2, C_3\}$, $\mathcal{K}(C_1) = \{Q_1, Q_2\}$, $\mathcal{K}(C_2) = \{Q_3\}$, and $\mathcal{K}(C_3) = \{Q_4\}$.

Ho and Lee [63] establish the following connection between the edges of a clique tree of a chordal graph and the set of minimal vertex separators in the graph: (i) for every edge (Q, Q') of the clique tree there is a minimal vertex separator S such that $S = Q \cap Q'$; (ii) for every minimal vertex separator S there is an edge (Q, Q') in the clique tree such that $S = Q \cap Q'$. Moreover, the number of times any given minimal vertex separator appears

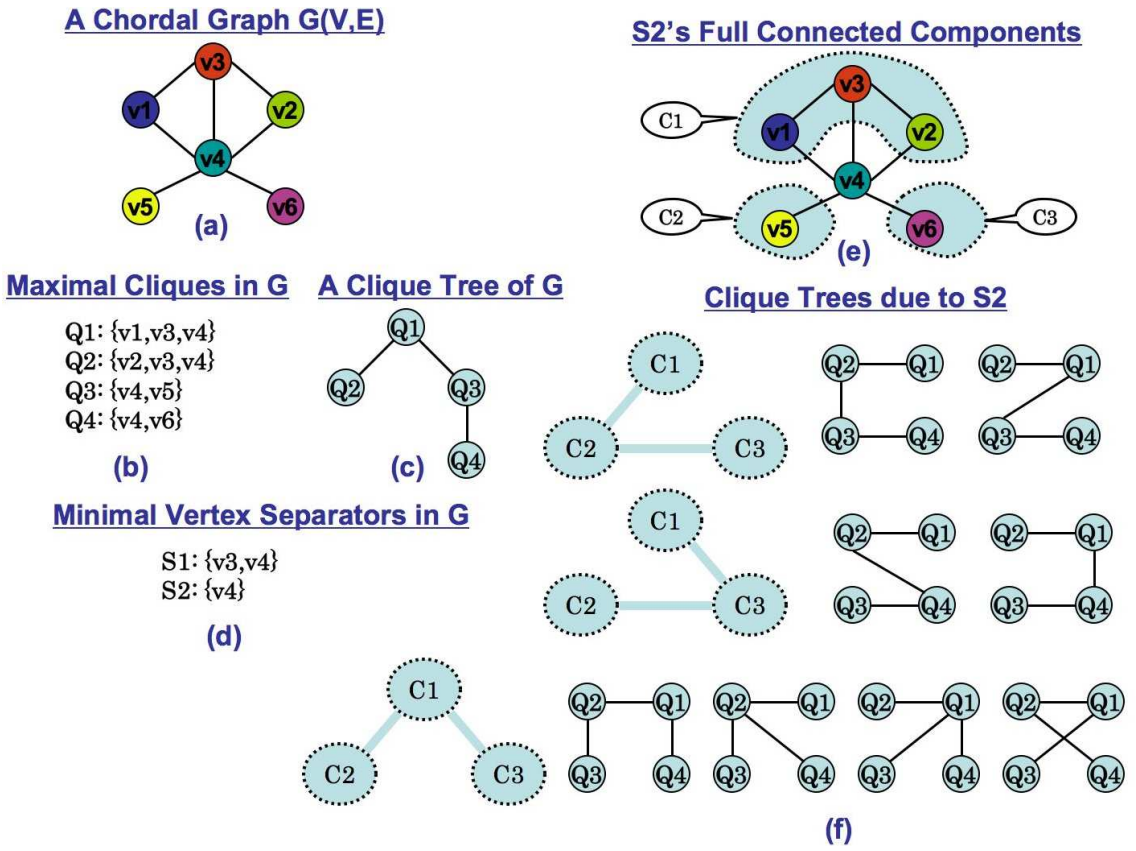


Figure 5.3: **(a)** A chordal graph $G = (V, E)$. **(b)** The set of maximal cliques in G. **(c)** A clique tree of G. **(d)** A set of minimal vertex separators of G. **(d)** The minimal vertex separator S_2 has three full connected components: C_1, C_2, and C_3. **(f)** The graph G has 8 different clique tree representations. In this case, all variations in the corresponding clique trees are due to the minimal vertex separator S_2 and are shown here. For each spanning tree of $\mathcal{C}(S_2)$, all possible realizations are shown. Thus, spanning tree that connects C_1 to C_2, and C_2 to C_3 has two different realizations.

in the multi-set $\{Q \cap Q' \mid (Q, Q') \text{ is an edge of the clique tree}\}$, is equal to the number of its full components minus one. For example, consider the chordal graph in Figure 5.3(a), the set of its minimal vertex separators in Figure 5.3(d), and one of its clique trees in Figure 5.3(c). The minimal vertex separator S_2 has three full connected components and therefore corresponds to two edges of the clique tree: (Q_1, Q_3) and (Q_3, Q_4).

Ho and Lee use the above connection to devise an algorithm, Algorithm 1, that given a chordal graph $G = (V, E)$ can generate any clique tree of the graph. The algorithm grows a clique tree by adding a set of $|\mathcal{C}(S)| - 1$ edges to the tree for every minimal vertex separator S of the graph. To choose this set of edges, the algorithm first selects the interconnection pattern between full connected components of S in line 3 (by constructing a spanning tree on $\mathcal{C}(S)$) and then "realizes" this interconnection pattern by choosing a set of edges to be added to the clique tree in line 5. In their paper, Ho and Lee prove that the algorithm not only produces a valid clique tree but is also able to generate any clique tree for the graph through appropriate choices in lines 3 and 5. The graph in Figure 5.3(a) has eight clique tree representations. In this case all clique trees are due to the minimal vertex separator S_2. Figure 5.3(f) shows how each clique tree is obtained, by showing each spanning tree on $\mathcal{C}(S_2)$ to the left and all its possible realizations to the right. For example, the first row of Figure 5.3(f) shows a spanning tree that connects C_1 to C_2 and C_2 to C_3. This spanning tree can be realized in two different ways, by connecting Q_3 to either Q_1 or Q_2 and connecting Q_3 to Q_4.

As can be seen from the algorithm, each minimal vertex separator S of the graph contributes to the total number of clique trees through two quantities: (i) the number of full connected components of S; (ii) for every full connected component C, the number of maximal cliques in $\mathcal{K}(C)$. Quantity (i) affects the number different interconnection

Algorithm 1 Generate a clique tree for a chordal graph

Require: A chordal graph G, a set of its maximal cliques, $\mathcal{Q}(G)$, and a set of its minimal separators, $\Delta(G)$.

Ensure: $T = (\mathcal{Q}(G), \mathcal{E})$ is a clique tree for G.

1: $\mathcal{E} \leftarrow \emptyset$

2: **for** every $S \in \Delta(G)$ **do**

3: Choose an arbitrary spanning tree T' on the set of full components of S, $\mathcal{C}(S)$

4: **for** every edge (C_1, C_2) in T' **do**

5: Set $\mathcal{E} \leftarrow \mathcal{E} \cup \{(Q_1, Q_2)\}$, where Q_1 is an arbitrary maximal clique from $\mathcal{K}(C_1)$ and Q_2 is an arbitrary maximal clique from $\mathcal{K}(C_2)$.

6: **end for**

7: **end for**

patterns of $\mathcal{C}(S)$ and the quantity (ii) affects the number of realizations of each such pattern. To be more precise, if $\mathcal{C}(S) = \{C_1, ..., C_k\}$, then according to Cayley's formula the number spanning trees on $\mathcal{C}(S)$ is equal to $\sum_{d_1+...+d_k=2k-2, d_i \geq 1} \binom{k-2}{d_1-1,...,d_k-1}$, where d_i denotes the degree of C_i in a given spanning tree. The number of different realization of a spanning tree with degree sequence $\{d_1, ..., d_k\}$ is equal to $\prod_i |\mathcal{K}(C_i)|^{d_i}$ and therefore the number of clique trees due to S is equal to

$$\sum_{d_1+...+d_k=2k-2, d_i \geq 1} \left(\binom{k-2}{d_1-1, ..., d_k-1} \prod_{i=1}^{k} |\mathcal{K}(C_i)|^{d_i} \right).$$

Ho and Lee use the above to show that the total number of clique trees is equal to

$$\prod_{S \in \Delta(G)} \left(\left(\sum_{C \in \mathcal{C}(S)} |\mathcal{K}(C)| \right)^{|\mathcal{C}(S)|-2} \prod_{C \in \mathcal{C}(S)} |\mathcal{K}(C)| \right).$$

5.1.2 Cographs and Modular Decomposition

Another graph theoretic technique that we use is *modular decomposition*. Consider a pair of nodes u and v that have exactly the same set of neighbors, i.e., $\mathcal{N}(u) \setminus \{v\} = \mathcal{N}(v) \setminus \{u\}$. We call the nodes of a pair *strong siblings* if they are connected by an edge, and *weak siblings* otherwise. The strong siblings relationship is a transitive relationship, meaning that if u and v are a pair of strong siblings and v and w are also a pair of strong siblings, then u and w are strong siblings as well. Thus, the strong siblings relationship is an equivalence relationship and a maximal set of strong siblings is well defined. The same holds for the weak siblings relationship.

Modular decomposition of a graph is obtained by iteratively contracting maximal sets of strong and weak siblings in the graph, until no more such sets can be found. At this point, the graph contains either a single node or an irreducible set of nodes, which is called a *prime module*. Thus, modular decomposition results in a tree-like hierarchical representation of the graph, where the leaf nodes are in one-to-one correspondence with the nodes of the graph and internal nodes correspond to contracted maximal sets of strong and weak siblings. Each contracted maximal set of strong siblings is replaced by a *series module* and each contracted maximal set of weak siblings is replaced by a *parallel module* (cf. Figure 5.4(a)-(e)).

While modular decomposition can be applied to any graph, only graphs that belong to a special graph family called *cographs* can be completely decomposed [26], i.e., the decomposition stops with a trivial prime module. When a graph can be completely decomposed, its modular decomposition tree can be used to derive a Boolean expression that describes all maximal cliques in the graph. The Boolean expression is constructed by moving along the tree from the leaves to the root, replacing each series module with an $\wedge$

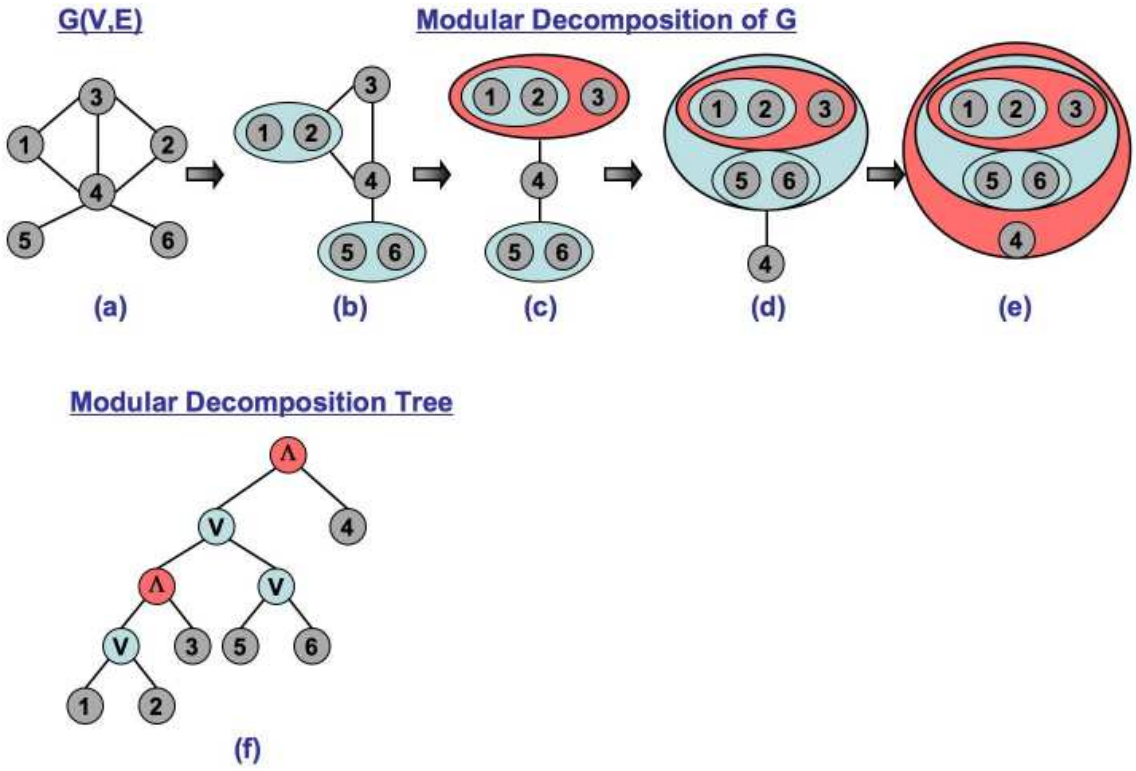


Figure 5.4: **(a)-(e)** Contraction steps taken during modular decomposition of a graph. **(b)**First, two maximal sets of weak siblings are contracted to supernodes $\{1,2\}$ and $\{5,6\}$. **(c)** Then, a maximal set of strong siblings is contracted to a supernode $\{\{1,2\},3\}$. **(d)** Then, a maximal set of weak siblings is contracted to a supernode $\{\{\{1,2\},3\},\{5,6\}\}$. **(e)** Finally, a maximal set of strong siblings is contracted to a supernode $\{\{\{\{1,2\},3\},\{5,6\}\},4\}$. **(f)** The corresponding modular decomposition tree. A Boolean expression that describes all the maximal cliques in the graph is $(((1 \vee 2) \wedge 3) \vee 5 \vee 6) \wedge 4$.

operator and every parallel module with an $\vee$ operator (cf. Figure 5.4(f)). An alternative characterization of cographs is in terms of forbidden subgraphs. A graph is a cograph if and only if it does not contain a path of length four (P_4), where the length is the number of nodes, as an induced subgraph. We will refer to this forbidden subgraph as an *induced* P_4 later in the text.

5.2 The Complex Overlap Decomposition Method

In this work we use graph-theoretic tools to identify pseudo-complexes from protein interactions within a functional module and to provide an alternative representation of the functional module. The main idea behind our method, which is depicted in Figure 5.5, is to provide a representation of a functional module that is analogous to a clique tree representation for chordal graphs, but in which nodes are cographs (representing pseudo-complexes) rather than maximal cliques (representing protein complexes). A pseudo-complex is either a protein complex or a set of alternative variants of such complex. For example, in the hypothetical protein interaction network in the upper-left corner of Figure 5.5, cliques $\{1, 2, 3\}$ and $\{1, 2, 4\}$ may correspond to two variants of one complex, where proteins 3 and 4 replace each other.

To systematically capture variants within a pseudo-complex, we use cographs to model pseudo-complexes. Recall from Section 5.1.2 that all maximal cliques in a cograph can be compactly represented by a Boolean expression. In the context of our application, this Boolean expression provides a compact representation of all variants of a protein complex within the corresponding pseudo-complex. Moreover, absence of an induced P_4 guarantees that the diameter of a connected cograph is at most two. Consequently, connected cographs are dense and cliquish, consistent with the assumption made by algorithms that

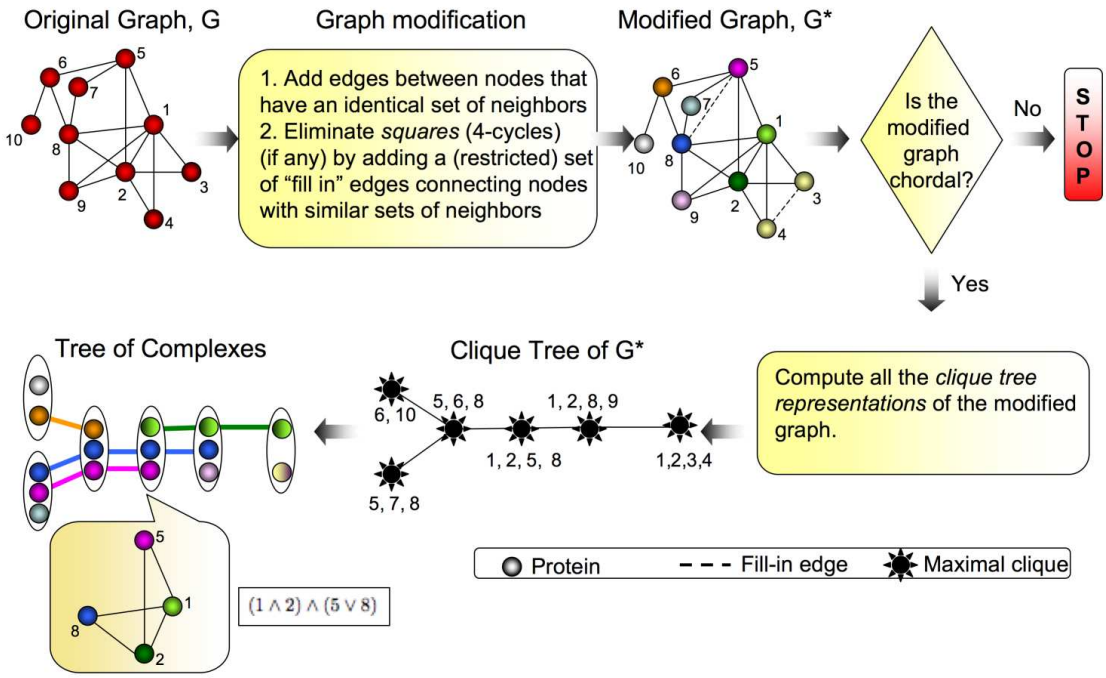


Figure 5.5: An illustration of the Complex Overlap Decomposition (COD) method. We add to the graph an edge, $(3, 4)$, connecting a pair of weak siblings. A fill-in edge between proteins 5 and 8 is added to eliminate all five 4-cycles in the graph: {5, 6, 8, 7}, {1, 5, 7, 8}, {2, 5, 7, 8}, {1, 5, 6, 8}, and {2, 5, 6, 8}. If the modified graph is chordal, all clique tree representations are computed and each such representation is extended into a Tree of Complexes representation of the original graph. The Tree of Complexes is constructed by projecting each maximal clique in the modified graph, G^*, to a pseudo-complex in the original graph G. For example, a four node maximal clique, $\{1, 2, 5, 8\}$, in G^* is projected to a four node pseudo-complex in G, by removing a fill-in edge $(5, 8)$. Each pseudo-complex is represented by a Boolean expression, such as $(1 \wedge 2) \wedge (5 \vee 8)$, which means that the pseudo-complex contains two variants of a complex, $\{1, 2, 5\}$ and $\{1, 2, 8\}$.

delineate protein complexes.

If we knew in advance all pseudo-complexes in the module, then we could simply connect the proteins within each pseudo-complex to turn it into a clique and, under the assumption that the resulting graph is chordal, apply a clique tree construction algorithm to the graph. Since we do not have predefined pseudo-complexes, our algorithm identifies them by adding edges to the graph in such a way that each added edge connects a pair of nodes that putatively belong to the same pseudo-complex.

The COD method's edge addition strategy and its biological motivation build on the concept of weak siblings. In terms of protein interaction networks, weak siblings are proteins that interact with the same set of proteins but do not interact with each other. In particular, proteins that can substitute for each other in a protein interaction network may have this property. Similarly, weak siblings may correspond to a pair of proteins that belong to the same complex but are not connected by an edge due to missing data or an experimental error. In both cases we would like any two such proteins to end up together in one or more pseudo-complexes. Thus, the COD method takes a first step towards delineation of pseudo-complexes by connecting every pair of weak siblings by an edge.

A pair of weak siblings can also be a source of multiple chordless cycles of length four or *squares* in the graph, where the length is the number of nodes. For example consider a pair of weak siblings, u and v. Pairs of non-adjacent nodes in the common neighborhood of u and v in the network together with u and v form squares. Therefore, connecting every pair of weak siblings the COD method not only delineates pseudo-complexes but also eliminates some of the squares in the graph.

If, after connecting all pairs of weak siblings, the resulting graph is not chordal,

the COD method attempts to transform it to chordal by adding some additional edges. Consistent with our assumption that we connect only nodes corresponding to proteins that could be put in the same pseudo-complex, we impose restrictions on this "fill-in" process. Namely, we require that each introduced edge connects a pair of nodes that are close to being weak siblings. In such a case the new edge is a diagonal of one or more squares in the protein interaction network. We emphasize that adding edges between nodes of longer cycles has no such justification. The edge addition procedure is described in more detail in Section 5.2.1.

If the modification step succeeds (i.e., the modified graph is chordal) all the clique tree representations of the modified graph are constructed and then extended to the Tree of Complexes representations of the original graph. The COD algorithm keeps track of all the edge additions and uses this information to delineate pseudo-complexes by projecting each maximal clique onto the original network and removing all introduced edges contained in the clique. For example, in the modified graph of Figure 5.5 a maximal clique with four nodes, $\{1, 2, 5, 8\}$, is projected to a pseudo-complex by removing an edge connecting protein 5 and 8. This pseudo-complex contains two variants of a protein complex, $\{1, 2, 5\}$ and $\{1, 2, 8\}$, which are compactly represented by the Boolean expression $(1 \wedge 2) \wedge (5 \vee 8)$. If, on the other hand, the modified graph is not chordal, the COD method stops without producing the representation.

5.2.1 Edge Addition Procedure

If after connecting every pair of weak siblings the resulting network is not chordal then the COD method attempts to eliminate the remaining squares by adding a limited set of edges that: (i) connect potentially functionally equivalent proteins, as measured by

the overlap in neighborhoods or distance from being a pair of weak siblings; (ii) ensure that a subgraph induced by the members of each pseudo-complex is a cograph.

We formulate the problem of finding a set of edges satisfying the above requirements as an optimization problem. Each set of edges, $S = \{e_1, ..., e_r\}$, is assigned a cost:

$$\texttt{cost}(S) = \sum_i (1.0 - \texttt{sim}(e_i)) \ ,$$

where $\texttt{sim}(e_i)$ takes values between 1.0 and 0.0, and measures our confidence in adding the edge to the graph. Since the addition of $e_i = (u_i, v_i)$ implies an interaction or functional equivalence between proteins u_i and v_i, we chose $\texttt{sim}(e_i)$ to be the amount of overlap between the neighborhoods of u_i and v_i, i.e., $\texttt{sim}(e_i) = \frac{|\mathcal{N}(u_i) \cap \mathcal{N}(v_i)|}{|\mathcal{N}(u_i) \cup \mathcal{N}(v_i)|}$, where $\mathcal{N}(v_i)$ denotes a set of neighbors of node v_i in the graph. Intuitively, $\texttt{sim}(e_i)$ measures how close u_i and v_i are to being a pair of weak siblings. If u_i and v_i have the same neighborhoods then $\texttt{sim}(e_i) = 1.0$; as the overlap between the neighborhoods decreases, $\texttt{sim}(e_i)$ goes to 0.0.

As described in Section 5.2.1.1 the COD method uses a reduction to the `Minimum Vertex Cover` problem to find all the minimal sets of up to k edges that eliminate all the squares in the graph, where minimal means that no proper subset of the set eliminates all the squares in the graph. From these sets it then picks an edge set with the minimum cost among all the sets that do not form an induced P_4 entirely contained in one of the maximal cliques of the modified graph. As shown in the following lemma the last requirement is necessary to ensure that a subgraph induced by the members of a pseudo-complex is a co-graph.

Lemma: For every pseudo-complex, a subgraph of the original graph induced by the members of the group contains an induced P_4 if and only if the set of edges added by our algorithm contains an induced P_4.

Proof: The argument follows from Figure 5.6. Indeed, (v_1, v_2, v_3, v_4) is a P_4 in the original

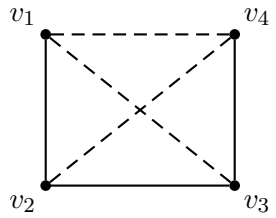


Figure 5.6: A P_4 in the subgraph induced by the members of a pseudo-complex corresponds to a P_4 in the set of added edges. Solid lines correspond to the original edges and dashed lines correspond to the added edges.

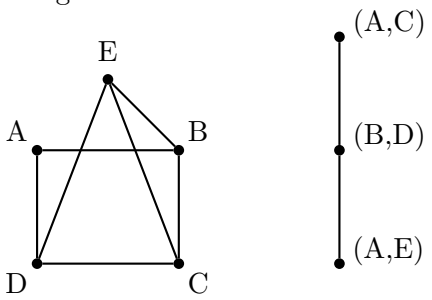


Figure 5.7: A graph and a corresponding "square coverage graph".

graph if and only if (v_3, v_1, v_4, v_2) is a P_4 formed by the added edges.

5.2.1.1 Reduction to the Minimum Vertex Cover

A square in a graph can be eliminated by adding one or both of its chords (diagonals) to the graph. For example, a graph in Figure 5.7 has two squares: (A, B, C, D) and (A, B, E, D). Note that (B, C, D, E) is not a square as one of its diagonals, (C, E), is an edge in the graph. The square (A, B, C, D) can be eliminated if either edge (A, C) or (B, D) is added to the graph, and the diagonal (B, D) eliminates both squares. We are interested in finding all minimal sets of diagonals of size up to k that eliminate all the squares in the graph.

We reduce the above problem to the `Minimum Vertex Cover` problem. The squares in the original graph become edges and diagonals become nodes in the new graph. Thus the original graph is transformed to a *square coverage graph*, which in turn serves as an input to the `Minimum Vertex Cover` problem. In the `Minimum Vertex Cover` problem

we are given a graph and are asked to find the smallest set of nodes that cover all the edges in the graph. An edge is covered if at least one of its end points is selected. Coming back to our example, it can be easily seen that $\{(B, D)\}$ is the minimum vertex cover (Figure 5.7). The minimum vertex cover in the square coverage graph will give us the minimum set of diagonals needed to eliminate all the squares in the original graph.

Although the `Minimum Vertex Cover` problem is an NP-hard problem, if the size of the optimum solution is small an efficient algorithm can be obtained. In other words the `Minimum Vertex Cover` problem is fixed-parameter tractable. We use an $O(2^k n)$ algorithm [32] to identify all minimal sets of edges of size up to k that eliminate all the squares in the graph.

5.3 Experimental Results

5.3.1 Mating Pheromone Signaling Pathway

In order to adapt to their environment, cells have to detect and respond to a vast variety of external stimuli. The detection and translation of these stimuli to a specific cellular response is achieved through a mechanism called *signal transduction pathway* or *signaling pathway*. The general principles of signal propagation through a pathway are common to almost all signaling pathways. First, an extracellular stimulus, usually a chemical ligand, binds to a membrane bound receptor protein. The energy from this interaction changes the state of the receptor protein, thus activating it. The active receptor is able to pass the signal to the *effector system* that generates the cell's response, for example through activation of a group of transcription factors and subsequent change in the expression of corresponding genes.

A variety of proteins carry information between the receptor protein and the effector

system, the most common being *protein kinases*. A protein kinase is a special enzyme that can add a phosphate group to certain residues of certain proteins through a process called *phosphorylation*. Phosphorylation changes the protein's ability to interact with other proteins, either activating or suppressing it, and therefore is analogous to turning a protein on or off.

The mating pheromone signaling pathway that we analyze here is one of the best studied signaling pathways. Our description of this pathway, its organization and components is based on a review by Bardwell [8]. There are two mating types of yeast cells. When a yeast cell is stimulated by a pheromone secreted by a cell of an opposite mating type, it undergoes a series of physiological changes in preparation for mating, which include significant changes in gene expression of about 200 genes, oriented growth towards the partner, and changes in the cell-cycle. Signal propagation through the pathway is achieved through interaction of the some 20 proteins. These interactions are schematically represented and described in Figure 5.8.

We have taken protein interactions that span pathway components from the DIP CORE network, a reliable subset of interaction from the DIP database [107]. The network is shown in Figure 5.9(a). Since proteins `STE2`/`STE3` are disconnected from the rest of the components, we have removed them from the network in our analysis. The COD method adds three diagonals, (`STE4`, `BEM1`), (`FUS3`, `KSS1`), and (`GPA1`, `STE5`), to eliminate eleven squares in the network, which results in twelve pseudo-complexes listed in Figure 5.9 along with the corresponding Boolean expressions. There are twelve Tree of Complexes representations for this protein interaction network. All the representations agree on the interconnection pattern between pseudo-complexes, B-E, H, and J-L. The difference between various tree variants comes from how pseudo-complexes A, F-G, and

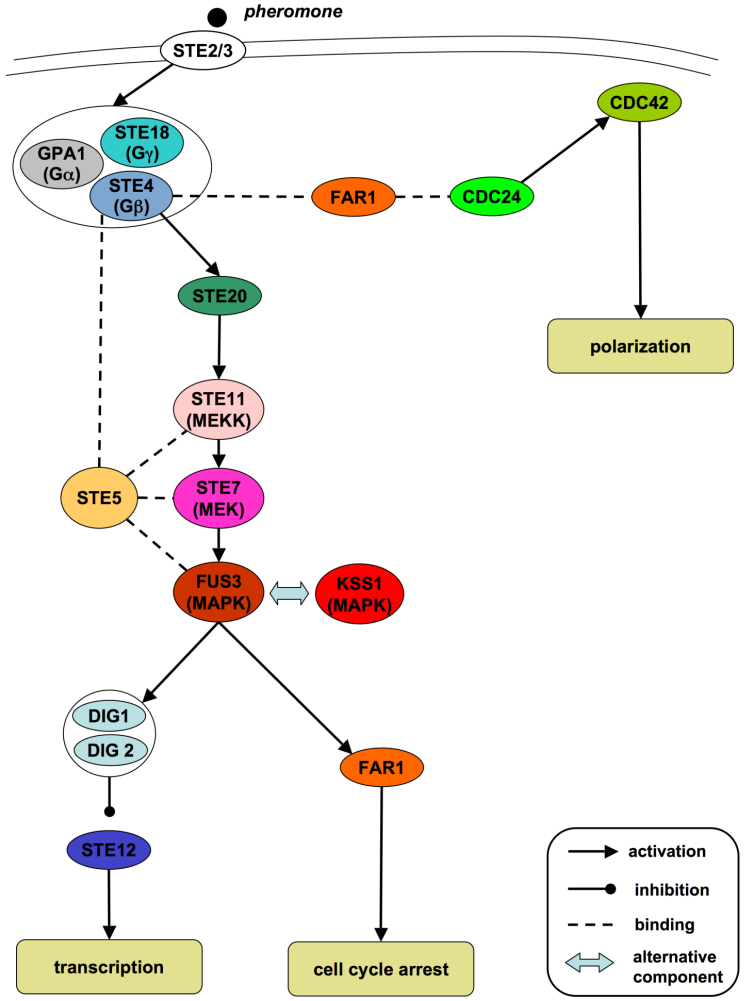


Figure 5.8: A schematic representation of the key components of the pheromone signaling pathway assembled from information in [8]. A pheromone peptide binds a G-protein coupled receptor or GPCR (STE2/STE3). The activated receptor binds and activates a trimeric G-protein: G_α subunit (GPA1), G_β subunit (STE4) and G_γ subunit (STE18). The flow of information then proceeds via a three-tiered mitogen-activated protein kinase (MAPK) cascade and results in activation of STE12 transcription factor and subsequent upregulation of about 200 genes. The MAPK cascade also activates the FAR1 protein, which is hypothesized to trigger a G_1 cell-cycle arrest through an interaction with CDC28, a master regulator of the cell-cycle. The MAPK cascade consists of three protein kinases STE11, STE7 and either FUS3 or KSS1, which activate each other sequentially through phosphorylation. Thus STE11 activates STE7, which in turn activates either FUS3 or KSS1. The phosphorylation process is enhanced through a presence of a scaffold protein STE5, which binds and thus co-localizes all three components of the MAPK cascade. Activated FUS3 and KSS1 proteins in turn bind their substrates, DIG1/DIG2/STE12 complex and FAR1 protein. Another branch of the pathway, which includes proteins FAR1, CDC24, CDC42, and BEM1, is responsible for triggering a "polarized growth towards the mating partner" or polarization response.

I are connected to the rest of the tree: (i) pseudo-complex A can be attached either through (A, C), or (A, B), or (A, J); (ii) pseudo-complex I through (I, E), or (I, D); (iii) pseudo-complexes F-G through (F, E) or (F, H).

In what follows we use the Tree of Complexes representation shown in Figure 5.9(b) but the same argument applies to other representations as well. The activation of the pathway corresponds to node A in the tree, which contains the G_β protein. From node A, the Tree of Complexes splits into two branches. One branch roughly corresponds to the MAPK cascade activated response, while another branch roughly corresponds to the morphogenesis response.

The MAPK cascade branch spans four nodes in the tree: I, D, E, and H. The `STE50` protein aids in activation of `STE11` by `STE20`, which in our representation comes out nicely in how I node merges with the rest of the MAPK branch. There is no interaction between `STE11` and `STE20` in the DIP network. As a result `STE20` is not a part of the MAPK branch of the tree. The activation of transcription factor complex by `FUS3` and `KSS1` is in nodes F and G. The morphogenesis branch spans nodes J, K and L.

Compare the representation in Figure 5.9(b) to the schematic representation of the pheromone signaling pathway shown in Figure 5.8. Using only protein interaction information, the COD method was able to recover two branches of the pathway, the MAPK cascade branch (I, D, E, H) and the polarization branch (J, K, L).

5.3.2 DNA Replication Module

DNA replication is a process by which cells duplicate their genetic material during cell division. To ensure that each daughter cell receives a complete and accurate copy of the DNA from the mother cell, each segment of the DNA has to be copied exactly once

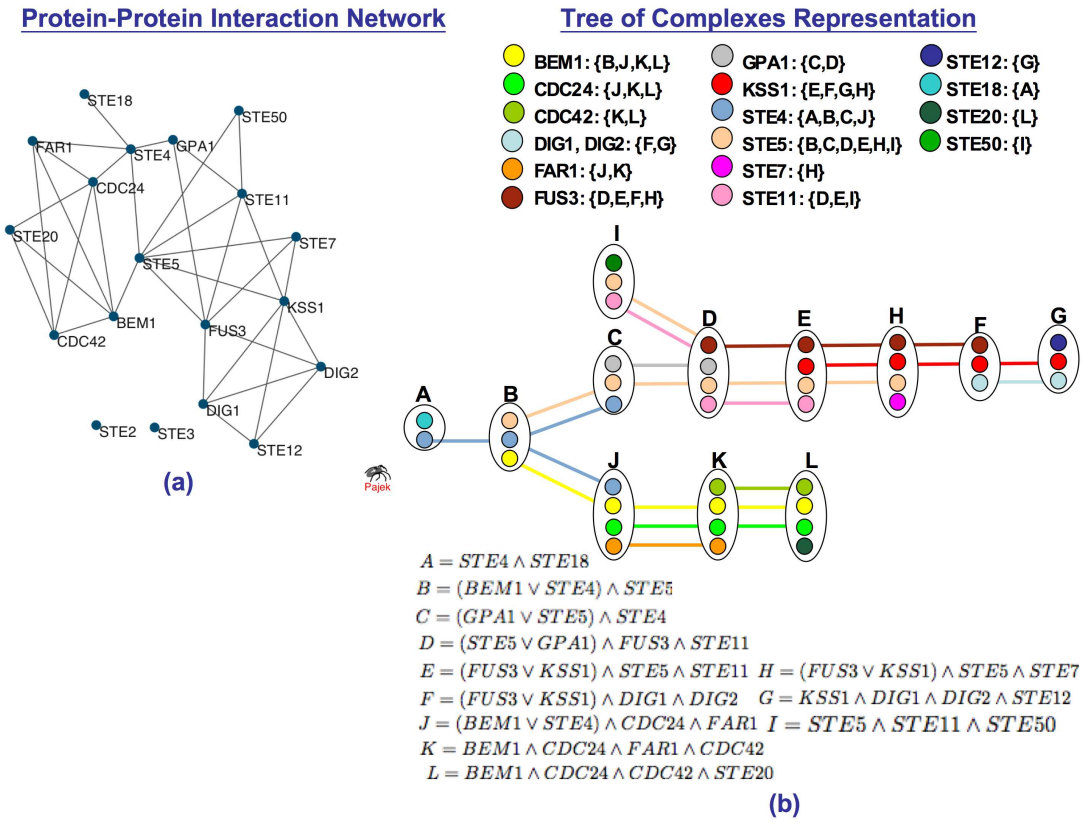


Figure 5.9: **(a)** The protein interaction network for the components of the pathway. The network was drawn with Pajek [11]. **(b)** One of the twelve possible Tree of Complexes representations for the network. The activation of the pathway corresponds to node A in the tree which contains the G_β (`STE4`) protein. From node A, the Tree of Complexes splits into two branches. One branch roughly corresponds to the MAPK cascade activated response, while another branch roughly corresponds to the polarization response. The MAPK cascade branch spans four nodes in the tree: I, D, E, and H. The activation of transcription factor complex by `FUS3` and `KSS1` is in nodes F and G. The polarization branch spans nodes J, K and L.

and the copying process has to be completed within a certain time window. To achieve this coordination, eukaryotic cells use a complex molecular machinery, where key protein complexes are formed through an ordered series of steps. Our description of these protein complexes and their components is adapted from an excellent review of eukaryotic DNA replication by Bell *et al.* [13].

In yeast, DNA replication starts at several sites (about 400) along the genome, termed *replication origins*, and proceeds in a parallel fashion, where each replication origin recruits molecular machinery needed to copy a segment of DNA to the neighboring replication origin.

The activation of origins occurs in two successive steps. First, a pre-replication complex (pre-RC) is assembled. The formation of pre-RC marks potential sites for the initiation of DNA replication and is commonly referred to as the origin licensing step. The activation of pre-RC complex by cyclin-dependent kinases (CDKs) and Dbf4-dependent kinases (DDKs) triggers formation of pre-initiation complexes (pre-ICs) around the origin that recruit molecular machinery necessary to duplicate the DNA. This molecular machinery includes following elements: DNA helicases are enzymes that separate DNA strands; ssDNAs proteins bind a single stranded DNA to prevent its entanglement; DNA polymerases are enzymes that synthesize a polynucleotide chain, selecting between four different nucleotides at each step according to the instructions of the complementary strand.

In yeast, pre-RC formation involves an ordered assembly of the following four proteins/protein complexes: ORC (Origin Recognition Complex), `CDC6`, `CDT1`, and MCM (Mini-Chromosome Maintenance) complex. Upon activation, proteins `CDC6` and `CDT1` leave the origin. The release of `CDC6` and `CDT1` coincides with the recruitment of `CDC45`

and a ssDNA, RPA1 (Replication Protein A). It is believed that CDC45 serves as a bridge between pre-RC and the proteins involved in DNA duplication: DNA Polymerase α, RFC (Replication Factor C) complex, PCNA (Proliferating Cell Nuclear Antigen) complex, and DNA polymerase ϵ.

We applied the COD method to the network assembled by Jansen *et al.* using a Bayesian Network approach and multiple sources of interaction evidence [72]. The subnetwork under consideration, shown in Figure 5.10**(a)**, contains only the proteins identified from the protein interaction network by Jansen *et al.* , and not all proteins involved in the process. The protein interaction network is chordal and does not contain weak siblings. Therefore no graph modifications are necessary. There are three Tree of Complexes representations of the protein interaction network: pseudo-complex A can be attached to the rest of the tree either through (A, B), or (A, C), or (A, D).

In what follows we use the Tree of Complexes representation shown in Figure 5.10**(b)** but the argument holds for the other representations as well. The activation of pre-RC complex, nodes A and B in the Tree of Complexes, and subsequent recruitment of DNA polymerases and other molecular machinery involved in DNA copying is clearly visible. Proteins POL2 and DPB2, which are part of DNA polymerase ϵ, appear in nodes C and D respectively. DNA polymerase α/primase (proteins POL1, POL12, PRI1, and PRI2), RFC complex (proteins RFC2 and RFC5), and PCNA (protein POL30) appear later on.

5.4 Summary

Mature experimental techniques exist that allow the inference of protein interactions, and recent proteomic studies used these and other technologies to characterize protein interactions among the components of many cellular processes. Even though protein

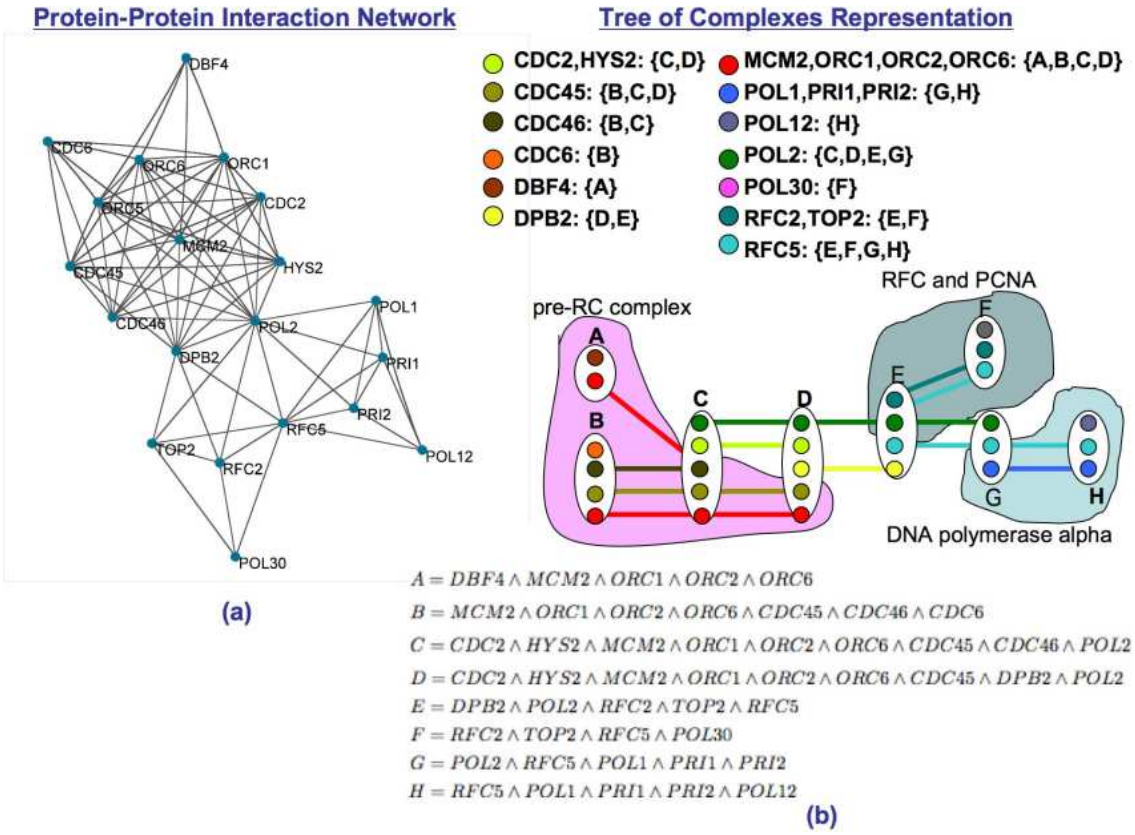


Figure 5.10: The DNA replication module. **(a)** Corresponding protein interaction network from the study of Jansen *et al.* [72]. **(b)** One of the three possible Tree of Complexes representation for the network.

interaction networks for many cellular processes are available, we have little knowledge of the dynamical properties of protein interactions involved in these processes.

To bridge this gap we developed a computational method to extract the dynamical properties of protein interactions from the inherently static topology of the protein interaction network. Given a protein interaction network spanning components of a cellular process, our method constructs a tree-like representation, called a Tree of Complexes representation, of the process. In this representation the nodes correspond to pseudo-complexes formed during the process. Moreover, the representation satisfies the additional condition that pseudo-complexes that contain any given protein induce a connected subgraph of the underlying tree. In this way, the representation captures not only the overlap between the pseudo-complexes but also the manner in which proteins enter and leave their enclosing pseudo-complexes. If the formation of pseudo-complexes during the process follows a specific order, our representation can be used to hypothesize about this order. Indeed, once the root of the tree is fixed, the representation induces a partial order between the pseudo-complexes in the module, which in turn can be used to infer temporal relationships.

The application of our method to two protein interaction networks underlying well studied cellular processes in *Saccharomyces cerevisiae* demonstrated that it is able to recover known temporal relationships.

Chapter 6

Topological Determinants of Lethality [3]

In their influential paper, Jeong *et al.* [73] observed that high-degree nodes in the protein interaction network of *Saccharomyces cerevisiae* are enriched in essential proteins. The authors further hypothesized that high-degree nodes tend to be essential due to the central role they play in maintaining the overall connectivity of the network by mediating interactions among other less connected proteins. Consequently, high-degree nodes are also referred to as *hubs* in the literature. (In this thesis we use the terms "high-degree nodes" and "hubs" interchangeably.)

The hypothesis of Jeong *et al.* implies that biological characteristics of a protein, such as lethality, may be explained by its placement in the network, i.e., topological prominence implies biological importance. If true, the hypothesis has important implications for the burgeoning field of Systems Biology.

In a recent study, however, He *et al.* [62] challenged the causal connection between global network topology and essentiality, and provided an explanation for the centrality-lethality rule in terms of *essential protein interactions*. Under the essential protein interactions model the majority of proteins are essential due to their involvement in one or more essential protein interactions that are distributed uniformly at random among the network edges. Consequently, hubs are predominantly essential because they are involved in more interactions and thus are more likely to be involved in one which is essential.

[3] This chapter is derived from "Essential complex biological modules explain the centrality-lethality rule" by E. Zotenko, J. Mestre, D. P. O'Leary and T. M. Przytycka, submitted for publication.

In this chapter we re-examine the connection between the topological prominence and essentiality. Toward this end, we conduct a rigorous analysis of five protein interaction networks for *Saccharomyces cerevisiae* compiled from diverse sources of interaction evidence; the networks used in this study are described in Section 6.4.1. We carefully evaluate the previously proposed explanations for the centrality-lethality rule on the tested networks. The results of this evaluation are described in Section 6.4.2 and Section 6.4.3.

There exist numerous measures of topological prominence, called *network centrality indices*; local centrality indices assign centrality values based on the topology of the node's local neighborhood whereas betweenness centrality indices assign centrality values based on the node's role in maintaining the connectivity between pairs of other nodes in the network. Even though by definition degree centrality is a local measure, depending on the structure of the network, hubs may play an important role in maintaining the overall connectivity of the network. To clarify the role of essential proteins in general and essential hubs in particular in maintaining the overall network connectivity, we compare degree centrality to other local and betweenness centrality indices. The centrality indices used in this study are described in Section 6.1.

If high-degree nodes play an important role in maintaining the overall network connectivity, then their removal should disrupt the connectivity between pairs of other nodes in the network as much as the removal of nodes having high betweenness centrality values. One common way to measure the impact of nodes' removal on the network connectivity is by monitoring the decrease in the size of the largest connected component. While the removal of a set of nodes may not disconnect various parts of the network, it may impair significantly the "quality of communication" between them. Therefore we introduce two additional measures, which we call *network integrity measures*, to capture various as-

pects of the effect of nodes' removal on the ability of other nodes to communicate. These measures are described in Section 6.2.

The results of our experiments indicate that the previously proposed explanations for the centrality-lethality rule do not hold in the tested networks. Therefore, we put forward an alternative explanation in terms of *essential complex biological modules*, abbreviated here as ECOBIMs. Essential complex biological processes are biological processes that are: (i) essential for an organism's vitality as measured by the large fraction of essential proteins and (ii) composed of proteins that interact extensively with each other. We hypothesize that the majority of hubs are essential due to their involvement in one or more ECOBIMs. To test our hypothesis we develop two complementary methods to extract putative ECOBIMs from a protein interaction network, described in Section 6.3. In Section 6.4.4 we demonstrate that membership in putative ECOBIMs accounts for the centrality-lethality rule in the tested networks.

6.1 Network Centrality Indices

A *network centrality index* assigns a centrality value to each node in the network and quantifies its topological prominence. Topological prominence can be defined in a number of ways, and over the years many centrality indices were introduced emphasizing different aspects of network topology [79]. In a local centrality index, the node's centrality value is mainly influenced by the topology of its local neighborhood. A well-known example of a local centrality index is degree centrality, where the node's centrality value is equal to the number of its immediate neighbors. Betweenness indices, on the other hand, assign centrality values based on the node's role in maintaining the connectivity between pairs of other nodes in the network. A well known example of a betweenness centrality index is

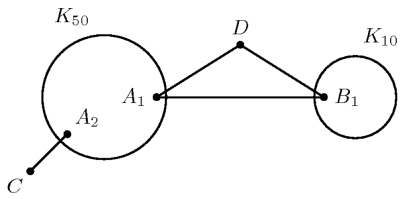
shortest-path betweenness centrality where the node's centrality value is proportional to the fraction of shortest paths that pass through it.

In this work we compare the *degree centrality* (DC) measure to two other local measures: *eigenvector centrality* (EC) [18] and *subgraph centrality* (SC) [33], and to two betweenness measures: *shortest-path betweenness centrality* (SPBC) [39], and *current-flow betweenness centrality* (CFBC) [93].

We give the precise definition of these measures in Sections 6.1.1-6.1.4 and we illustrate the differences among the five centrality measures on a toy network in Figure 6.1(a). In this network two cliques K_{50} and K_{10} are interconnected by an edge (A_1, B_1) and through a node D. The nodes of K_{50} are labeled $A_1 \ldots A_{50}$ and the nodes of K_{10} are labeled $B_1 \ldots B_{10}$. The additional node C attaches to K_{50} through A_2. Figure 6.1(b) shows the ranking of network nodes based on the centrality values assigned by the five centrality measures.

In the description of the centrality indices below we use n to denote the number of nodes, m the number of edges, and A the adjacency matrix of the protein interaction network under consideration. As the networks we deal with are undirected and unweighted, the adjacency matrix is a symmetric n-by-n 0-1 matrix such that $a_{ij} = 1$ if and only if the nodes i and j are adjacent.

Both the eigenvector and subgraph centrality indices rely on the *eigenvalue decomposition* of the adjacency matrix [117]: $A = U\Lambda U^T$, where $U = [\vec{u}_1, ..., \vec{u}_n]$ is an orthogonal matrix whose columns contain the right eigenvectors of A and $\Lambda = diag(\lambda_1, ..., \lambda_n)$ is a diagonal matrix of the eigenvalues of A; thus, we have $A\vec{u}_i = \lambda_i \vec{u}_i$. We will further assume that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$.



(a)

DC	EC	SC	SPBC	CFBC
A1	A1	A1	A1	A1
A2	A2	A2	B1	B1
A3...A50	A3...A50	A3...A50	A2	D
B1	B1	B1	A3...A50, B2...B10, C, D	A2
B2...B10	D	D		B2...B10
D	C	C		A3...A50
C	B2...B10	B2...B10		C

(b)

Figure 6.1: The difference between centrality measures demonstrated on a toy network. **(a)** The toy network consists of two cliques K_{50} with nodes $A_1 \dots A_{50}$ and K_{10} with nodes $B_1 \dots B_{10}$. The two cliques are interconnected by an edge (A_1, B_1) and through an additional node D. Additional node C attaches to the network through A_2. **(b)** Ranking of network nodes according to centrality values produced by the five measures.

6.1.1 Eigenvector Centrality

The eigenvector centrality index assigns centrality values based on the eigenvector that corresponds to the largest eigenvalue of the adjacency matrix of the network. The derivation of the eigenvector centrality values can be cast in a form of an iterative process: (i) start with an initial vector of centrality scores $\vec{c}_0 = (c_1^0 \dots c_n^0)$, (ii) in iteration $k+1$ update the centrality score of a node i using the scores of its neighbors from the previous iteration: $c_i^{k+1} = \sum_{j \text{ is a neighbor of } i} c_j^k$, and then normalize the scores $\vec{c}_{k+1} = \frac{\vec{c}_{k+1}}{||\vec{c}_k||_2}$. In matrix form this is expressed as $\vec{c}_{k+1} = \frac{A\vec{c}_k}{||\vec{c}_k||_2}$.

It can be shown that the above iterative process converges to the eigenvector that corresponds to the largest eigenvalue of the adjacency matrix of the network. In fact, this procedure is equivalent to the widely used *power method* for computing the largest eigenvalue of a matrix and its corresponding eigenvector [97]. Thus, the centrality values

are the entries in the vector $\vec{u}_1$.

6.1.2 Subgraph Centrality

The subgraph centrality value of a node is proportional to the number of closed walks that start and terminate at the node. The number of walks of length k that start and terminate at a given node is given by the diagonal entries of A^k and the number of closed walks of any length by the diagonal entries of $\sum_{k=0}^{\infty} A^k$. To ensure finite centrality values, the number of closed walks of length k is weighted by $\frac{1}{k!}$; i.e., the centrality values are equal to the diagonal elements of $\sum_{k=0}^{\infty} \frac{A^k}{k!}$.

The centrality values can be efficiently computed using the eigenvalue decomposition of A. Indeed, if $A^k = U\Lambda^k U^T$ then $\sum_{k=0}^{\infty} \frac{A^k}{k!}$ is equal to $U(\sum_{k=1}^{\infty} \frac{\Lambda^k}{k!})U^T$ or $U\,diag(e^{\lambda_1}, ..., e^{\lambda_n})U^T$.

6.1.3 Shortest-Path Betweenness Centrality

Under the shortest-path betweeness index, the node's centrality value is equal to the average fraction of shortest paths that pass through the node. Let us denote by $\sigma_{s,t}$ the number of shortest paths between nodes s and t, and by $\sigma_{s,t}(i)$ the number of shortest paths between s and t that pass through a third node i. Then the centrality value of the node i is equal to $\sum_{s,t} \frac{\sigma_{s,t}(i)}{\sigma_{s,t}}$.

The straightforward computation of shortest-path betweenness values requires $\Theta(n^3)$ time and $\Theta(n^2)$ space. In this thesis we use the algorithm due to Brandes [20] that allows computation of the centrality values in $O(nm + n^2 \log n)$ time and $O(n + m)$ space.

6.1.4 Current-Flow Betweenness Centrality

The current-flow centrality measure extends the shortest-path centrality measure by taking into account other paths in addition to shortest paths. This is achieved using a current flow paradigm, where the network is viewed as an electrical current network with each edge having a unit resistance. When a unit of current is introduced at a source node, s, and is removed at a sink node, t, current flows along the paths from s to t with shorter paths getting bigger amounts of flow. The current-flow centrality value of a node is equal to the total amount of current that passes through the node summed over all possible pairs of source and sink nodes.

Given a pair of source and sink nodes, s and t, computing the amount of current that passes through other nodes of the graph involves solving for voltage values, $\vec{v}_{s,t} = \{v_1^{s,t}, ..., v_n^{s,t}\}$, that satisfy the *Kirchhoff's potential (voltage) law*:

$$\text{for each } i, \quad \sum_{j \text{ is adjacent to } i} v_i^{s,t} - v_j^{s,t} = b_i^{s,t} = \begin{cases} 0 & : \quad i \neq s \text{ and } i \neq t \\ +1 & : \quad i = s \\ -1 & : \quad i = t \end{cases},$$

where $|v_i^{s,t} - v_j^{s,t}|$ is the amount of current that flows between i and j. If $v_i^{s,t} - v_j^{s,t} > 0$ then the current flows from i to j, and if $v_i^{s,t} - v_j^{s,t} < 0$ then the current flows from j to i.

The above constraints form a system of linear equations: $(D - A)\vec{v}_{s,t} = \vec{b}_{s,t}$. The matrix $(D - A)$ is the *Laplacian* of the graph, one of several special matrices associated with graphs. It is well known that the Laplacian corresponding to a connected graph with n nodes has rank $(n - 1)$; thus the above system has an infinite number of solutions. Indeed, if $\vec{v}_{s,t}$ is a solution then so is $\vec{w}_{s,t} = \vec{v}_{s,t} + a\vec{\mathbf{1}}$ for any scalar a. Therefore, in [93] a unique solution $\vec{v}_{s,t}$ is obtained by additionally requiring that $\vec{v}_r^{s,t} = 0$ for an arbitrary

node r; for simplicity let $r = n$.

Let W denote a matrix obtained from $(D_n - A_n)^{-1}$ by adding a zero last column and row:

$$W = \begin{pmatrix} (D_n - A_n)^{-1} & \mathbf{0}_{(n-1)\times 1} \\ \mathbf{0}_{1\times(n-1)} & 0 \end{pmatrix},$$

where D_n and A_n denote matrices obtained from D and A by deleting the n_{th} row and column. Pre-computing W allows for efficient computation of voltage values for all pairs of source and sink nodes. Indeed, given a pair of source and sink nodes, s and t, $\vec{v}_{s,t} = W\vec{b}_{s,t} = \vec{w}_s - \vec{w}_t$, and thus can be computed in $O(n)$ time.

Given voltage values $\vec{v}_{s,t}$, the current-flow betweenness centrality value of i is equal to $\sum_{s,t}\sum_{j \text{ is adjacent to } i} |v_i^{s,t} - v_j^{s,t}|$. Computing the current-flow betweenness values takes $O(n^2m)$ time where computing W takes $O(n^3)$ and summarizing the voltage values takes $O(n^2m)$ time.

Current-flow betweenness centrality is by far the most computationally expensive index among the tested indices. Therefore, its implementation had to be fine-tuned to make computation of centrality values for large networks feasible. As we implemented the centrality indices using the Python programming language, we had to be careful to avoid loop constructs and to reduce the code to matrix/vector computations that are implemented efficiently in the Python numerical package `numpy`. For example, for a network with $2,316$ nodes and $5,569$ edges the computation of W takes 28 seconds, the summarization step using a straightforward implementation takes over 2 hours, and the summarization step using the fine-tuned implementation takes 14 minutes. The pseudo-code for straightforward and fine-tuned implementations is given in Algorithm 2 and Algorithm 3.

Algorithm 2 Compute current-flow betweenness centrality values

Require: The matrix W.

Ensure: $\vec{c}$ is a vector whose entries are current-flow betweenness values.

1: $\vec{c} \leftarrow \vec{\mathbf{0}}$

2: **for** every s in $1...n$ **do**

3: **for** every t in $s+1...n$ **do**

4: Set $\vec{v} \leftarrow \vec{w}_s - \vec{w}_t$.

5: **for** every edge $e = (i, j)$ **do**

6: Update $c_i \leftarrow c_i + |v_i - v_j|$.

7: Update $c_j \leftarrow c_j + |v_i - v_j|$.

8: **end for**

9: **end for**

10: **end for**

Algorithm 3 Compute current-flow betweenness centrality values

Require: The matrix W.

Ensure: $\vec{c}$ is a vector whose entries are current-flow betweenness values.

1: $\vec{c} \leftarrow \vec{\mathbf{0}}$

2: **for** every i in $1...n$ **do**

3: Form $\hat{W}_{d_i \times n}$ from rows of W that correspond to the neighbors of i.

4: Form $R_{d_i \times n}$ such that $r_{j,t} = w_{i,t} - \hat{w}_{j,t}$.

5: **for** every s in $1...n$ **do**

6: Form $X_{d_i \times n}$ such that $\vec{x}_t = \vec{r}_s - \vec{r}_t$.

7: Update $c_i \leftarrow c_i +$ sum of the absolute values of entries of X. (Correctness follows from the fact that the value of c_i is increased by $\sum_t \sum_{j \text{ is adjacent to i}} |v_i^{s,t} - v_j^{s,t}| = \sum_t \sum_{j \text{ is adjacent to i}} |(w_i^s - w_j^s) - (w_i^t - w_j^t)|$.)

8: **end for**

9: **end for**

6.2 Network Integrity Measures

We introduce two measures, which we call *network integrity measures*, to capture various effects of node removal on the ability of other nodes to communicate. An integrity measure maps a set of nodes S to a value between 0 and 1, with the value of 0 being assigned when the removal of S completely disrupts the communication and the value of 1 being assigned when it causes no disruption.

In the description of the integrity measures below we use $G(V, E)$ to denote the protein interaction network under consideration. We use n to denote the number of nodes and m the number of edges in the network.

6.2.1 Shortest-Path Integrity

Our first measure, shortest-path integrity, quantifies the increase in the length of the shortest path due to the removal of S and is given by $\frac{\sum_{s,t \notin S} \max(C - d_S(s,t), 0)}{\sum_{s,t \notin S} \max(C - d(s,t), 0)}$, where $d(s,t)$ is the length of the shortest path between s and t in the original network, $d_S(s,t)$ is the length of the shortest path between s and t after the removal of S, and C is a constant. In this work we set the value of C to be twice the diameter of the original network.

6.2.2 Edge-disjoint Paths Integrity

Our second measure, edge-disjoint paths integrity, quantifies the decrease in the number of edge-disjoint paths and is given by $\frac{\sum_{s,t \notin S} f_S(s,t)}{\sum_{s,t \notin S} f(s,t)}$, where $f_S(s,t)$ is the number of edge-disjoint paths between s and t in the modified network and $f(s,t)$ is this value in the original network.

Computing the number of edge-disjoint paths between two nodes, s and t, amounts to a max-flow computation. Thus, the naive approach to computing the edge-disjoint

paths integrity measure takes $O(n^2\,T(n,m))$ time, where $T(n,m)$ is the time it takes to find a maximum flow in an undirected unweighted graph. (The currently-known best value for $T(n,m)$ is $O(\min(n^{3/2},m)m^{1/2})$, which is due to Goldberg and Rao [52].) However, there is a more efficient way to compute the integrity measure using a number of graph theoretical concepts such as Menger's Theorem, Gomory-Hu Trees and Min-Weight Tree Decomposition.

Given a pair of nodes s and t, an s-t cut is a partition of nodes in the network into two sets X and Y containing s and t respectively. The cost of the cut (X,Y) is defined as the number of edges that cross the boundary between X and Y. A minimum s-t cut is defined as an s-t cut of minimum cost. Clearly the maximum number of edge-disjoint s-t paths cannot exceed the cost of the minimum s-t cut. Menger's Theorem [86] states that these two quantities are, in fact, equal. Thus, we can rewrite our integrity measure as $\frac{\sum_{s,t\notin S} c_S(s,t)}{\sum_{s,t\notin S} c(s,t)}$, where $c(s,t)$ is the cost of a minimum s-t cut in G and $c_S(s,t)$ is the cost of a minimum s-t cut in $G[V\setminus S]$. This quantity can be computed with the aid of a Gomory-Hu Tree. For simplicity we describe how to compute $\sum_{s,t\notin S} c(s,t)$.

In their classical 1961 paper, Gomory and Hu [54] introduced the notion of a *cut tree*, also known as *Gomory-Hu tree*, which succinctly encodes the value $c(s,t)$ for all $s,t\in V$. A cut tree is a weighted tree $\mathcal{T}$ with node set V. For any pair of nodes $s,t\in V$, let $m(s,t)$ be the minimum weight edge in the unique path connecting s and t in $\mathcal{T}$, and let $X_{m(s,t)}$ and $Y_{m(s,t)}$ be the node sets of the two trees obtained by removing $m(s,t)$ from $\mathcal{T}$. For any $s,t\in V$, a cut tree has the following remarkable properties: (i) the weight of $m(s,t)$ equals $c(s,t)$; (ii) the cut $(X_{m(s,t)},Y_{m(s,t)})$ is a minimum s-t cut. Not only did Gomory and Hu prove that a cut tree always exists, but they also showed how to compute a cut tree by performing $n-1$ max-flow computations. Although the high level idea behind

their algorithm is simple, its implementation is rather involved. For our purpose we use a much simpler algorithm due to Gusfield [57] to compute an *equivalent flow tree*, which also runs in $O(n\,T(n,m))$ time. An equivalent flow tree only has the first propety of a cut tree described above, namely, the weight of $m(s,t)$ equals $c(s,t)$ for any $s,t \in V$.

Denote by $\mathcal{T}$ the equivalent flow tree of G. The naive way to compute $\sum_{s,t\notin S} c(s,t)$ using $\mathcal{T}$ takes $O(n^3)$ time since finding $m(s,t)$ for $s,t \notin S$ can take $\Theta(n)$ time in the worst case. A more efficient alternative is to use a *min-weight tree decomposition* of $\mathcal{T}$. Given a weighted tree $\mathcal{T}$, its min-weight tree decomposition is a rooted full binary tree $\mathcal{D}(\mathcal{T})$ whose internal nodes are edges of $\mathcal{T}$ and leaf nodes are nodes of $\mathcal{T}$. The tree $\mathcal{D}(\mathcal{T})$ is defined recursively as follows:

- If $\mathcal{T}$ has a single node u, then $\mathcal{D}(\mathcal{T})$ consists of the single node u.
- Otherwise, $\mathcal{T}$ has at least one edge. Let e be a minimum weight edge of $\mathcal{T}$ and let $\mathcal{T}_1$ and $\mathcal{T}_2$ be the two trees obtained from $\mathcal{T}$ by removing e. Then the root of $\mathcal{D}(\mathcal{T})$ is e and its two children are the roots of $\mathcal{D}(\mathcal{T}_1)$ and $\mathcal{D}(\mathcal{T}_2)$ respectively.

Given an equivalent flow tree $\mathcal{T}$ of a graph, it is easy to compute $\sum_{s,t\notin S} c(s,t)$ in $O(n)$ time using its min-weight tree decomposition. Indeed, each internal node e in $\mathcal{D}(\mathcal{T})$ contributes to $\sum_{s,t\notin S} c(s,t)$ its weight times the number of pairs $s,t \notin S$ such that s is a leaf in the subtree rooted at the left child of e and t is a leaf in the subtree rooted at the right child. The computation of $\mathcal{D}(\mathcal{T})$ can be done in $O(n^2)$ time by following its recursive definition, or in expected $O(n \log n)$ time using a randomized algorithm [87].

6.3 Computational Methods for Identifying Essential Complex Biological Modules

We hypothesize that the majority of hubs are essential due to their involvement in essential complex biological processes (ECOBIMs), biological processes that are indispensable for organism's vitality and whose components interact extensively with each other. Therefore, in general, ECOBIMs correspond to highly connected subnetworks of nodes, with shared biological function, that are enriched in essential proteins. Proteins are deemed to share biological function if they are annotated with the same GO biological process term from a set of 192 biological process terms, selected by a group of experts to represent relevant aspects of molecular biology [92].

To test our hypothesis, we developed two complementary methods for automatic extraction of putative ECOBIMs from a protein interaction network. Both methods were applied to subnetworks induced by proteins annotated with the same biological process GO term, one network at a time. The high-level idea underlying the methods is to start from highly connected seeds of proteins and iteratively add nodes maintaining high connectivity. Thus, the methods start with a seed, a k-clique of proteins, and extend it through addition of proteins that have at least r neighbors already in the seed; the result is the set of putative ECOBIMs returned by the methods. We should mention that for a given GO subnetwork all possible seeds are explored, as a result several, possibly overlapping ECOBIMs may be extracted from the subnetwork.

The main difference between the methods is in the way the enrichment in essential proteins is achieved. The first method is a one-step procedure where the enrichment and high connectivity are enforced simultaneously by requiring that: (i) initial seeds are k-cliques of essential proteins, and (ii) a non-essential protein is considered for addition only if it is adjacent to at least r_{ess} essential proteins already in the seed.

The second method, on the other hand, is a two-step procedure where the enrichment in essential proteins is achieved through a filtering step that follows the initial seed selection step. In the filtering step the minimal number of seeds that cover most of the essential proteins initially present is greedily selected. Let us denote by $\mathcal{S} = \{S_1, ..., S_l\}$ the set of initial seeds and by $\mathcal{E}$ the set of essential proteins present in $\cup_{i=1}^{l} S_i$. Ideally we would like to select the cheapest subset of initial seeds that cover a large enough fraction of proteins in $\mathcal{E}$, where the cost of the seed is equal to the fraction of non-essential proteins that it contains. This is precisely the *Partial Set Cover* problem. Define the benefit of a seed $S \in \mathcal{S}$ with respect to a given collection $\mathcal{C} \subseteq \mathcal{S}$ as the number of essential proteins in S that do not belong to any set in $\mathcal{C}$. Slavik [114] studied a simple heuristic that greedily builds a solution by picking at each step a seed, minimizing its cost divided by its benefit with respect to the sets chosen so far, until the coverage requirement is met. He showed that this algorithm produces a solution with cost at most $\ln \Delta$ times the optimum, where Δ is the maximum number of essential proteins in any seed. On the negative side, Fiege [36] showed that for any constant $\epsilon > 0$ there is no polynomial time algorithm that returns a solution with cost at most $(1 - \epsilon) \ln \Delta$ times the optimum unless NP problems can be solved in quasi-polynomial time. Therefore, Slavik's result is essentially the best possible. In our application $\Delta \approx 150$; thus the heuristic is guaranteed to return a solution no worse than 5 times the optimum; in practice, however, the solutions found are usually much better than what the worst case analysis guarantees.

6.4 Experimental Results

6.4.1 Protein Interaction Networks

Recently several hypotheses that linked structural properties of protein interaction networks to biological phenomena have come under scrutiny [3, 17, 27, 10, 9], with the main concern being that the observed properties are due to experimental artifacts and/or other biases present in the networks and as such lack any biological implication. To limit the impact of such biases on the results reported in our study, we selected five genome-wide protein interaction networks for *Saccharomyces cerevisiae* compiled from diverse sources of interaction evidence, as described in Section 4.2: a high-confidence network derived mostly from small-scale experiments (the DIP CORE network) [29], a network derived solely from small-scale studies reported in the literature (the LC network) [101], a high-confidence network derived from a variety of interaction sources (the HC network) [10], a network derived solely from high-throughput affinity purification experiments (the TAP-MS network) [25], and a network derived solely from interactions predicted in silico using Bayesian formalism (the BAYESIAN network) [72].

Table 6.1 summarizes structural properties of the tested networks. (Here and throughout this chapter we work with the largest connected component of each protein interaction network.) The networks differ not only in the number of nodes/edges but also in number of other structural parameters. For example, TAP-MS and BAYESIAN networks are much more cliquish than the other networks, judging by the number of cliques present in the network (data not shown). This is not surprising as the edges in these networks correspond to membership in multi-protein complexes.

	ess.	nodes	edges	degree	distance	k-conn.	3-cliques	5-cliques
DIP CORE	0.29	2,316	5,569	4.81	5.22	2.16	0.50	0.12
LC	0.27	3,224	11,291	7.00	4.22	2.77	0.59	0.24
HC	0.30	2,752	9,097	6.61	4.90	2.81	0.60	0.26
TAP-MS	0.32	1,994	15,819	15.87	4.82	4.74	0.72	0.51
BAYESIAN	0.22	4,135	20,984	10.15	4.33	3.13	0.44	0.22

Table 6.1: Structural properties of the protein interaction networks used in our study: fraction of essential proteins, number of nodes, number of edges, average degree, average shortest path, average number of edge disjoint paths, fraction of nodes covered by 3-cliques, and fraction of nodes covered by 5-cliques.

6.4.2 Lethality and Betweenness

Even though degree centrality is a local centrality index, in some networks hubs may play an important role in maintaining the overall connectivity of the network. For example, it was demonstrated that in some scale-free networks the removal of hubs affects the ability of other nodes to communicate much more than the removal of random nodes [1]. To clarify the topological role of hubs in the tested networks, we compared degree centrality to two other local indices (eigenvector centrality (EC) [18] and subgraph centrality (SC) [33]), and to two betweenness indices (shortest-path betweenness centrality (SPBC) [39], and current-flow betweenness centrality (CFBC) [93]).

Since betweenness indices rank nodes based on their role in mediating communication between pairs of other nodes in the network, it is interesting to see whether hubs are as effective in disconnecting the network as nodes with high betweenness centrality values.

One common way to measure the impact of nodes' removal on the network connectivity is by monitoring the decrease in the size of the largest connected component. Figures 6.2(a)-(e) show, for the five protein interaction networks, how the removal of the most central nodes, random nodes, and essential proteins affects the network connectivity. As expected, removing nodes with high local centrality values is much less disruptive than removing those with high betweenness centrality values. Interestingly, degree centrality is

	shortest-path integrity					
	dc	ec	sc	spbc	cfbc	rand
DIP CORE	3.31e-03	3.03e-01	2.14e-01	9.39e-03	2.19e-03	7.88e-01±2.85e-02
LC	1.11e-02	3.92e-01	3.87e-01	7.47e-02	3.24e-03	8.43e-01±4.28e-02
HC	1.36e-01	5.67e-01	5.91e-01	6.02e-02	5.23e-03	8.24e-01±3.43e-02
TAP-MS	4.45e-01	6.51e-01	6.51e-01	5.34e-02	3.30e-02	8.07e-01±2.49e-02
BAYESIAN	1.81e-01	6.22e-01	6.05e-01	9.56e-02	5.19e-03	NA

	edge-disjoint paths integrity					
	dc	ec	sc	spbc	cfbc	rand
DIP CORE	2.23e-03	3.15e-01	2.10e-01	7.08e-03	1.68e-03	6.89e-01±2.54e-02
LC	1.08e-02	3.82e-01	3.75e-01	8.19e-02	2.34e-03	7.43e-01±2.88e-02
HC	1.41e-01	5.33e-01	5.58e-01	6.65e-02	4.03e-03	7.21e-01±2.19e-02
TAP-MS	3.21e-01	5.49e-01	5.49e-01	7.21e-02	5.11e-02	7.39e-01±2.20e-02
BAYESIAN	1.98e-01	5.77e-01	5.67e-01	1.05e-01	1.86e-02	NA

Table 6.2: Two network integrity measures, shortest-path integrity and edge-disjoint paths integrity, are used to quantify the impact of the removal of the 20% most central nodes on the network connectivity. An integrity measure maps a set of nodes S to a value between 0 and 1, with the value of 0 being assigned when the removal of S completely disrupts the communication and the value of 1 being assigned when it causes no disruption. We also show the impact of node removal in random order. These values for the BAYESIAN network are not available (shown as NA), as their computation is computationally demanding.

as efficient in shattering the network as betweenness in the DIP CORE and LC networks, is as inefficient as the local indices in the TAP-MS network, and is somewhere in between the local and betweeness indices in the HC and BAYESIAN networks.

While the removal of a set of nodes may not disconnect various parts of the network, it may impair significantly the "quality of communication" between them. For example, there can be an increase in the length of shortest path or decrease in the number of alternative paths between pairs of nodes in the network. Network integrity measures capture various aspects of the effect of nodes' removal on the ability of other nodes to communicate. (See Section 6.2 for the description of the network integrity measures.) We find that even when these more sensitive measures are used, the observations made above about the disruptive power of hubs relative to other most central proteins hold (see Table 6.2).

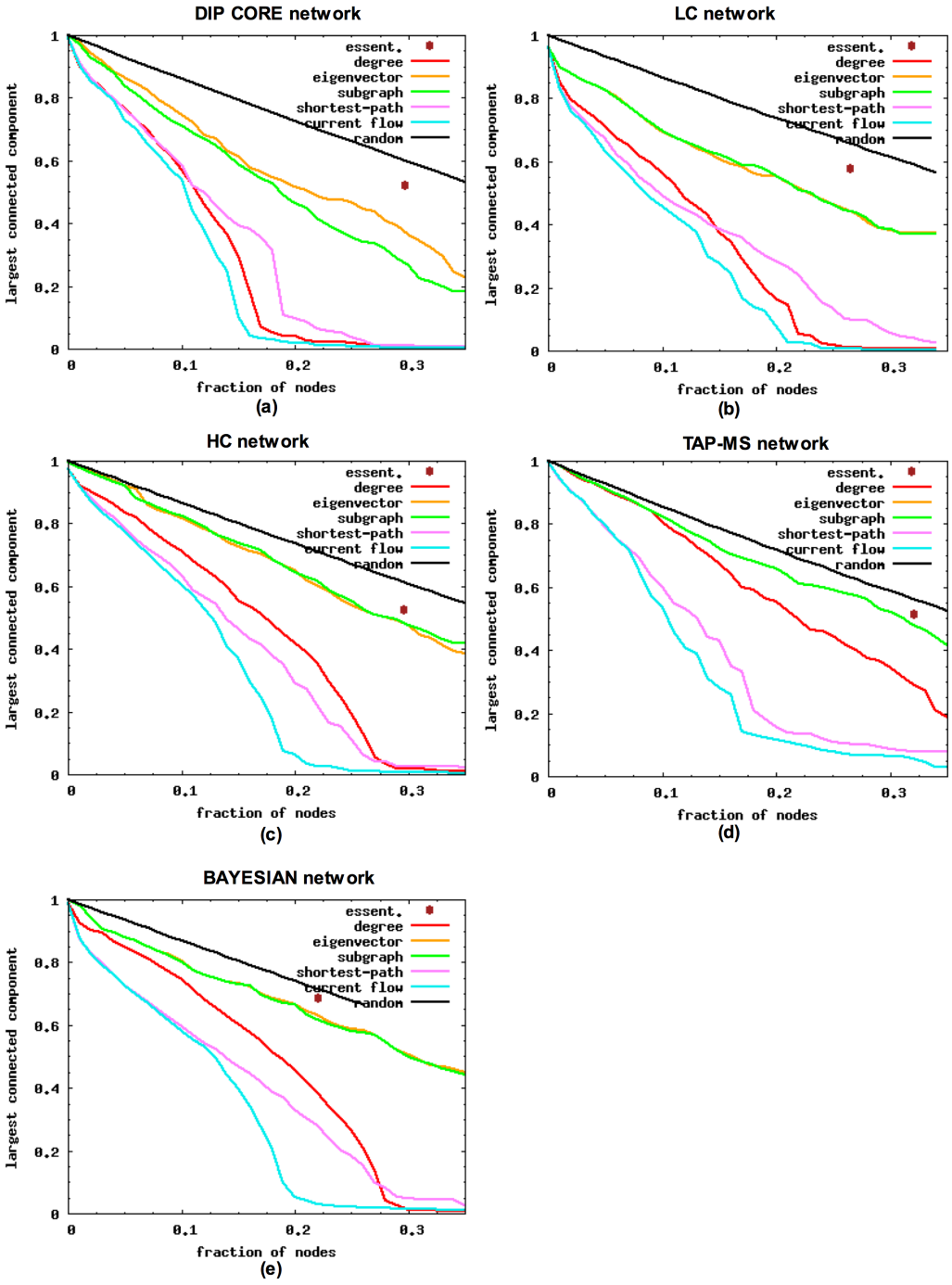


Figure 6.2: **(a)-(e)** The impact of node removal is quantified by the fraction of nodes in the largest connected component. There is one curve for each centrality measure that shows the fraction of nodes in the largest connected component as a function of the fraction of the most central nodes removed. We also show the impact of node removal in a random order and the size of the largest connected component when all essential proteins are removed.

	essential	random non-essential
DIP CORE	0.519	0.504 ± 0.007
LC	0.578	0.551 ± 0.010
HC	0.521	0.525 ± 0.005
TAP-MS	0.512	0.512 ± 0.011
BAYESIAN	0.685	0.625 ± 0.006

Table 6.3: For each network, we compare the effect of the removal of essential proteins to the removal of an equivalent number of random non-essential proteins with the same degree distribution, by looking at the fraction of nodes in the largest connected component.

Next, we examined whether the disruption power of hubs comes mainly from essential hubs. First, we observe that the removal of all essential genes is less disruptive than the removal of an equivalent number of most central nodes according to any index (see Figure 6.2(a)-(e)). Moreover, as shown in Table 6.3, the removal of essential nodes is not more disruptive than the removal of an equivalent number of random non-essential nodes that have the same degree distribution. We conclude that even though in some networks, most notably in the DIP CORE, LC, and HC networks, the removal of hubs is disruptive, this disruption is not related to the essentiality of hubs. On the contrary, essential genes are indistinguishable in that respect from the random non-essential genes with the same degree distribution.

Above we demonstrated that various centrality indices vary considerably in their ability to measure disruption in the overall connectivity of the network. Next we asked whether this difference is reflected in the enrichment levels. Figure 6.3 shows the fraction of essential proteins among central proteins, taking the top 20% according to the five centrality indices. We observe that the local centrality indices have enrichment levels comparable to those of betweeness indices and in some cases even higher. But most notably, degree centrality fares better than any other centrality index in all five networks; the superiority of degree centrality is even more apparent when the Kendall's tau rank correlation

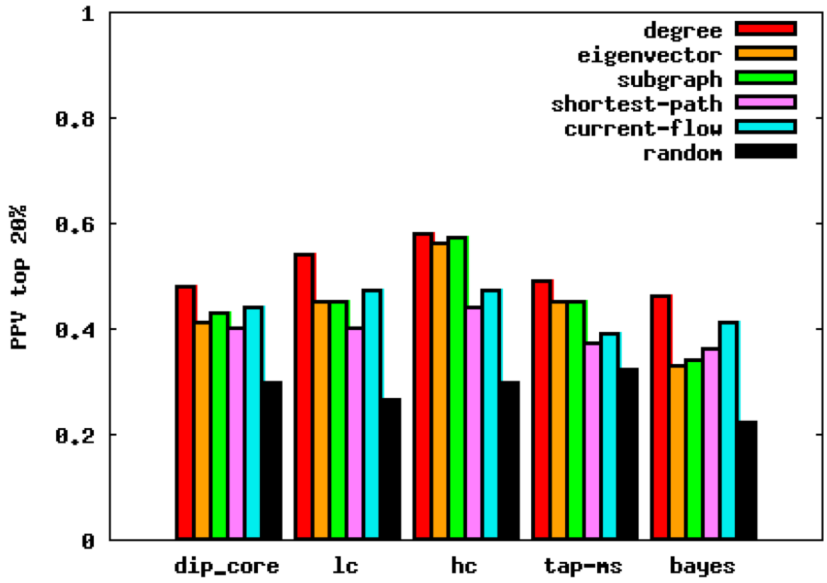


Figure 6.3: Fraction of essential proteins among the 20% most central proteins according to the five centrality measures; under random we show the expected fraction of essential proteins were the nodes drawn uniformly at random from the network nodes.

coefficient is used to measure correlation between centrality values and essentiality (see Table 6.4).

As there is considerable correlation between degree centrality and other centrality indices (see Table 6.4), we used Kendall's tau partial rank correlation coefficient to see whether any of the indices is correlated with essentiality beyond their correlation with degree centrality index. We found that, controlling for the correlation with degree, the correlation with essentiality is reduced to statistically insignificant values for betweenness centrality indices and is greatly reduced for local indices (see Table 6.4).

The above observations indicate that the main topological determinant of essentiality is the node's local neighborhood rather than its role in maintaining the overall connectivity of the network. In particular, even though betweeness centrality indices are much more effective in shattering some networks, their correlation with essentiality is reduced to statistically insignificant levels by subtracting their correlation with degree

	eigenvector			subgraph		
	τ_{dc}	τ_{ess}	$\tau_{ess.dc}$	τ_{dc}	τ_{ess}	$\tau_{ess.dc}$
DIP CORE	0.436 (29.6)	0.151 (8.9)	0.064 (3.8)	0.579 (39.3)	0.173 (10.2)	0.060 (3.5)
LC	0.461 (37.3)	0.225 (15.6)	0.095 (6.6)	0.462 (37.4)	0.225 (15.6)	0.094 (6.5)
HC	0.476 (37.6)	0.239 (15.4)	0.106 (6.8)	0.496 (37.2)	0.240 (15.4)	0.100 (6.4)
TAP-MS	0.516 (33.5)	0.117 (6.4)	-0.007 (0.4)	0.516 (33.5)	0.117 (6.4)	-0.007 (0.4)
BAYESIAN	0.466 (42.7)	0.165 (13.0)	0.046 (3.6)	0.467 (42.9)	0.170 (13.4)	0.051 (4.0)

	shortest-path			current-flow		
	τ_{dc}	τ_{ess}	$\tau_{ess.dc}$	τ_{dc}	τ_{ess}	$\tau_{ess.dc}$
DIP CORE	0.713 (46.2)	0.153 (8.6)	-0.002 (0.1)	0.836 (55.2)	0.188 (10.8)	0.013 (0.7)
LC	0.667 (51.7)	0.212 (14.1)	0.002 (0.1)	0.829 (65.4)	0.257 (17.4)	-0.008 (0.5)
HC	0.623 (45.0)	0.201 (12.4)	0.005 (0.3)	0.772 (56.7)	0.242 (15.2)	-0.006 (0.4)
TAP-MS	0.459 (28.5)	0.124 (6.5)	0.018 (0.9)	0.619 (39.5)	0.160 (8.6)	0.017 (0.9)
BAYESIAN	0.637 (56.5)	0.176 (13.4)	0.005 (0.4)	0.806 (72.1)	0.228 (17.6)	0.018 (1.4)

Table 6.4: We use Kendall's tau rank correlation coefficient to measure the correlation of centrality measure with degree centrality (τ_{dc}), with essentiality (τ_{ess}), and with essentiality after controlling for correlation with degree centrality ($\tau_{ess.dc}$). Statistical significance is assessed using z-scores which are shown in parentheses.

centrality.

6.4.3 Lethality and the Essential Protein Interactions Model

Recently He and colleagues [62] proposed an explanation for the centrality-lethality rule in terms of essential protein interactions: a protein is essential either due to its involvement in one or more essential protein interactions or due to other factors. The authors argue that the determination of protein essentiality in the protein interaction network can be captured by a simple random process: (i) distribute essential protein interactions among the edges of the network uniformly at random with probability α; (ii) distribute essential proteins among the nodes of the network uniformly at random with probability β. Thus, according to the model, the probability (P_E) of a protein with k neighbors being essential is $P_E = 1 - (1-\alpha)^k(1-\beta)$, and the natural logarithm of the fraction of non-essential proteins among proteins of degree k has a linear dependency on k: $\log(1-P_E) = \log(1-\alpha)k + \log(1-\beta)$.

	simulation		weighted line fitting		line fitting	
	α	β	α	β	α	β
DIP CORE	0.0649	0.0814	0.0286	0.2106	0.0626	0.0982
LC	0.0512	0.0255	0.0154	0.2097	0.0360	0.1456
HC	0.0662	0.0045	0.0304	0.1737	0.0375	0.1705

Table 6.5: We use three strategies to estimate the parameters, α and β, of the essential protein interaction model [62]: the network simulation as described in the original paper (simulation), line fitting to points $(\log(1-P_E), k)$ for $k \leq k_0$ (line fitting), and weighted line fitting to points $(\log(1-P_E), k)$ for all values of k (weighted line fitting).

To evaluate the model on the tested networks we used three strategies to estimate the model's parameters: a network simulation procedure, line fitting to points $(\log(1-P_E), k)$ for $k \leq k_0$, and weighted line fitting to points $(\log(1-P_E), k)$ for all values of k. (In weighted line fitting the contribution of $(\log(1-P_E), k)$ to the error function is weighted by the fraction of nodes having degree k.) The first two strategies are described by He *et al.* They deem the agreement of parameter values estimated using the network simulation and line fitting strategies to be one of the strongest indications for the validity of the model. But in our networks, the parameter values estimated using different strategies vary considerably (see Table 6.5). Moreover for estimates with high values of α the model results in a significantly higher fraction of essential proteins among high-degree nodes, as shown in Table 6.6. (In their paper, He *et al.* point out that their model may not work in networks where the edges represent membership in the same protein complex. Thus we excluded the TAP-MS and BAYESIAN networks from the analysis.)

We note that from the assumptions of the essential protein interaction model it follows that if two proteins do not interact then the essentiality of one protein in such a pair does not depend on the essentiality of the other protein. Furthermore, this independence should also be observed when proteins share interaction neighbors. To test whether this holds in real data, we computed the number of non-adjacent protein pairs with three or

	observed	expected		
		simulations	weighted line fitting	line fitting
DIP CORE	0.504	0.682 ($< 1.0e-05$)	0.513 ($4.2e-01$)	0.676 ($< 1.0e-05$)
LC	0.579	0.716 ($< 1.0e-05$)	0.472 ($1.0e-00$)	0.649 ($4.5e-03$)
HC	0.619	0.780 ($< 1.0e-05$)	0.594 ($8.1e-01$)	0.653 ($1.4e-01$)

Table 6.6: The difference between fraction of essential proteins among 10% highest degree nodes in real networks and that predicted by the essential protein interaction model. The statistical significance of the difference is measured with p-values which are shown in parenthesis.

	total	observed	expected		
			simulations	weighted line fitting	line fitting
DIP CORE	1,849	1,135	945.33 ($3.6e-10$)	936.22 ($5.2e-11$)	944.25 ($3.0e-10$)
LC	10,777	6,143	5,690.46 ($6.1e-10$)	5,553.41 ($7.9e-16$)	5,542.31 ($2.3e-16$)
HC	5,907	3,516	3,214.04 ($2.2e-08$)	2,969.04 ($5.4e-24$)	3,003.80 ($2.6e-21$)

Table 6.7: The number of non-adjacent protein pairs with three or more common neighbors where both proteins are either essential or non-essential. For each network, we show the total number of non-adjacent pairs with three or more common neighbors in the network (total), the number of pairs with both proteins being essential or non-essential in the network (observed), and the expected number under the model (expected) for three sets of model parameter values. In parentheses we show the statistical significance of the difference in observed and expected values estimated using the exact Fischer test.

more neighbors that are either both essential or both non-essential in the tested networks and compared these numbers to the expected number of such pairs under the model. As shown in Table 6.7, the model does not capture the correlation in essentiality observed in the tested networks, as there is a statistically significant difference between the number of such pairs observed in real data and the number expected under the model. Consequently, the essential interaction model is rejected with high confidence.

6.4.4 Lethality and Essential Complex Biological Modules

From the argument presented in Section 6.4.3 it follows that the essentiality of pairs of proteins that share neighbors is correlated. Therefore we hypothesized that densely connected subnetworks are either enriched or depleted in essential proteins in general

and in essential hubs in particular. Moreover, it is well known that densely connected subnetworks are also enriched in proteins that share biological function. In this section we demonstrate the existence of essential complex biological modules (ECOBIMs), highly connected subnetworks of nodes with shared biological function, which are enriched in essential proteins. (In this work proteins are deemed to share biological function if they are annotated with the same GO biological process term from a set of 192 terms which were selected by a group of experts to represent relevant aspects of molecular biology [92].) We show that most essential hubs belong to such ECOBIMs and the fraction of essential proteins among hubs that are not the members of ECOBIMs is much lower than what would be expected by chance.

We designed two complementary methods, described in Section 6.3, to extract putative ECOBIMs from a protein interaction network. For the DIP CORE, LC, and HC networks the values of parameters are $k = 4$, $r = 3$, and $r_{ess} = 2$. For the BAYESIAN and TAP-MS networks, due to the cliquish nature of these networks, the values of parameters are $k = 5$, $r = 4$, and $r_{ess} = 3$.

Although the methods differ in how the putative ECOBIMs are computed, they produce ECOBIMs that contain an almost identical set of proteins (see Table 6.8) implying that the heuristic applied in the second method captures the properties of ECOBIMs correctly. From now on we limit our analysis to the set of ECOBIMs identified by the first method, as the largely non-overlapping nature of this set makes the statistics more transparent.

The putative ECOBIMs identified by our two-step procedure mostly correspond to large essential multi-protein complexes such as *anaphase promoting complex (APC)* and *DAM1 protein complex*, but not exclusively complexes. For example, the largest ECOBIM

	one-step procedure				two-step procedure				
	num.	genes	ess. genes	enrich.	num.	genes	ess. genes	enrich.	overlap
DIP CORE	51	315	228	0.72	33	352	252	0.72	0.83
LC	77	790	513	0.65	33	815	516	0.63	0.80
HC	78	687	477	0.69	48	696	498	0.72	0.86
TAP-MS	44	620	375	0.60	31	611	386	0.63	0.87
BAYESIAN	70	764	492	0.64	37	825	513	0.62	0.88

Table 6.8: The putative ECOBIMs produced by the two methods agree on the set of proteins they contain. For each network we show the number of ECOBIMs identified by the two methods, and the number of genes and the number of essential genes they contain. We also show the amount of overlap between the corresponding set of genes, where the amount of overlap between sets A and B is $\frac{|A \cap B|}{|A \cup B|}$.

identified in the LC network contains multi-protein complexes involved in the process of RNA polymerase 2 transcription [59], such as RNA polymerase 2, general transcription factors, the mediator complex, etc.

To examine to what extent the membership in ECOBIMs accounts for the centrality-lethality rule we partitioned the top 20% of the nodes, ordered by degree, into two groups: those that are members of one or more ECOBIMs (ECOBIM hubs) and those that are not (non-ECOBIM hubs), and compared their enrichment values. As shown in Figure 6.4, the enrichment values for non-ECOBIM hubs are not only lower than those for ECOBIM hubs but are also lower than the background enrichment values.

We next asked whether there is a correlation between degree and lethality for network nodes that are not members of the ECOBIMs. Ideally, if the centrality-lethality phenomenon is completely accounted for by membership in ECOBIMs there would be no statistically significant correlation. Unfortunately the set of automatically identified ECOBIMs is an approximation only. For example, some proteins may be missing from their respective ECOBIMs due to incomplete functional annotation of the yeast proteome. As these proteins will generally have many neighbors among successfully identified ECOBIM members, we would expect the correlation, if it exists, to be due to edges that connect

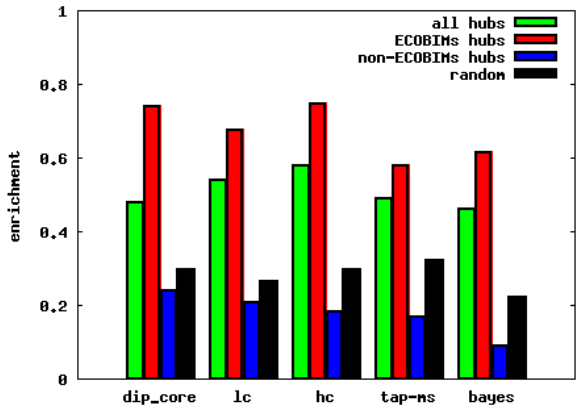


Figure 6.4: Fraction of essential proteins among various types of degree hubs, the 20% highest-degree nodes in the network: all hubs, hubs that are members of ECOBIMs (ECOBIM hubs), hubs that are not members of ECOBIMs (non-ECOBIM hubs). Fraction of essential proteins among all proteins in the network is also shown (random).

these proteins to the ECOBIMs. To check if this the case, we used the Kendall's tau rank correlation coefficient to compute correlation between essentiality and degree, essentiality and the number of ECOBIM neighbors, and partial correlation between essentiality and degree, controlling for correlation with the number of ECOBIM neighbors. As shown in Table 6.9, even though correlation between essentiality and degree for nodes that are not members of ECOBIMs is much less than that for all network nodes, it is still statistically significant. However, in all networks except the LC network, this correlation is reduced to statistically insignificant values after correction for correlation with the number of ECOBIM neighbors is performed.

6.5 Summary

The enrichment of high-degree nodes in essential proteins, known as the *centrality-lethality rule*, suggested that the topological prominence of a protein in a protein interaction network may be a good predictor of its biological importance. Even though the correlation between degree and essentiality was confirmed by many independent studies,

	all nodes	non ECOBIM nodes		
	τ_{dc}	τ_{dc}	$\tau_{ecobimdc}$	$\tau_{dc.ecobimdc}$
DIP CORE	0.217 (12.0)	0.078 (3.9)	0.169 (7.9)	0.035 (1.8)
LC	0.315 (20.8)	0.097 (5.4)	0.083 (4.4)	0.064 (3.6)
HC	0.318 (19.5)	0.066 (3.4)	0.161 (7.8)	0.001 (0.1)
TAP-MS	0.238 (12.6)	0.035 (1.5)	0.035 (1.4)	0.023 (1.0)
BAYESIAN	0.271 (20.3)	0.041 (2.7)	0.053 (3.3)	0.021 (1.4)

Table 6.9: We use Kendall's tau rank correlation coefficient to measure correlation between essentiality and degree (τ_{dc}), number of ECOBIM neighbors ($\tau_{ecobimdc}$), and degree controlled for correlation with the number of ECOBIM neighbors ($\tau_{dc.ecobimdc}$). For comparison we include correlation between essentiality and degree for all network nodes. Statistical significance of correlation values is given by z-scores which are shown in parentheses.

until recently there was no systematic attempt to examine the reasons for this correlation.

To identify the main topological determinant of essentiality and to provide a biological explanation for the connection between the network topology and essentiality, we performed a rigorous analysis of five genome-wide protein interaction networks for *Saccharomyces cerevisiae*. We demonstrated that the majority of hubs are essential due to their involvement in Essential Complex Biological Modules, a group of densely connected proteins that are annotated to the same biological process GO term, and are enriched in essential proteins. Moreover, we rejected two previously proposed explanations for the centrality-lethality rule, one relying on the assumption that essential hubs maintain the overall network connectivity and another relying on the recently published essential protein interactions model.

Chapter 7

Conclusions and Directions for Future Work

As the number of completely sequenced genomes grows we are faced with the important but daunting task of assigning function to proteins encoded by newly sequenced genomes. The main focus of this thesis was on developing computational methods which can be used to facilitate protein function assignment at the molecular and cellular levels.

One of the oldest and most powerful approaches for functional annotation of newly discovered proteins transfers annotation from evolutionarily related proteins of known function. Despite advances in protein sequence comparison, distantly related proteins can still only be detected by the most accurate protein structure alignment methods. Due to inherent difficulty of the protein structure alignment problem these methods are computationally expensive and cannot be used in high-throughput studies of protein structure. To overcome this deficiency, reliable methods are usually combined with less accurate but fast protein structure comparison methods. In Chapter 3 we described our contribution in the area of fast protein structure comparison.

Over the past decade several high-throughput experimental techniques to detect protein interactions were developed. These experimentally-determined interactions are routinely represented by a graph, a *protein interaction network*, with nodes representing the proteins and edges representing the interactions between the proteins. The study of the topological properties of these networks has become an important tool in studying protein function at the cellular level and formulating hypotheses about the general organization

principles of biological systems. In Chapter 5 and Chapter 6 we described our contribution in exploring the connection between the topology of protein interaction networks and protein biological function.

Here we summarize our contributions and present directions for future work.

7.1 Fast Protein Structure Comparison with Structural Footprinting

Projection methods are a class of fast protein structure comparison methods that achieve a considerable speed-up over full-fledged protein structure alignment methods by mapping a protein structure to a high-dimensional vector. Once the mapping is done the structural similarity is approximated by the distance computation between the corresponding vectors. In the process of mapping some structural information is lost. Thus, the central issue in designing a good projection method is how to define a mapping that is able to capture all the salient features of protein structure.

In Chapter 3 we systematically addressed this issue by introducing the structural footprinting framework. Our framework defines a family of projection methods that differ in the "structural alphabet" used by the method to describe protein structure. In fact, a large variety of methods can be generated that emphasize different aspects of protein structure.

We demonstrated that structural footprinting is a useful approach for designing fast protein structure comparison methods. In particular, the SSEF method is more accurate in detecting homologous protein pairs than other projection methods. Moreover, the results of our experiments indicate that the SSEF method is well suited to be combined with a full-fledged protein structure alignment method to allow high-throughput protein structure comparison. First, the method is extremely fast. Second, when a certain reasonable

number of errors is permitted, the method achieves coverage not only significantly higher than that of sequence comparison and other projection methods, but also comparable to that of some full-fledged protein structure alignment methods. We stress that the coverage at reasonably high error levels determines the suitability of the method for a screening application, since a few false positive results, which result in overhead for the method being sped up, can be tolerated as long as most of the related (similar) domains are retrieved.

We also explored how the retrieval accuracy of a structural footprinting method depends on the structural alphabet used. Not surprisingly, the SSEF method, whose structural alphabet incorporates secondary structure information and completely ignores less conserved loop regions, has the best performance on average in retrieving evolutionarily related protein pairs. However, we also found that no structural footprinting method performs the best in all cases, which means that some groups of evolutionarily related proteins exhibit structural variability that is better tolerated by structural alphabets used by LFF and SEGF methods. To take advantage of the relative strengths of the methods we proposed strategies to combine the methods. As expected the combined method significantly outperforms the SSEF method which is the best structural footprinting method. Moreover, the results of our experiments indicate that combining a pair of methods whose performance is least correlated results in the biggest improvement. Thus, combining the SSEF and SEGF methods is more beneficial than combining the SSEF and LFF methods, even though the LFF method has a significantly better performance than the SEGF method.

The results of our study point to promising directions for future work. We have shown the benefit of combining structural footprinting methods that employ complementary structural alphabets. Therefore, a systematic way of defining complementary

structural alphabets should be investigated. To expand the space of possible structural alphabets, inclusion of information based on additional aspects of protein structure should be investigated. In this thesis, structural alphabets were derived solely from the geometry of protein structure (i.e., the atomic coordinates) and no other information was used. For example, it was recently demonstrated that a structural descriptor of the third hypervariable (V3) loop region of the HIV viral gene coding for the envelope protein gp120, which combines structural information with physico-chemical properties of the corresponding residues, allows significantly better discrimination between the two co-receptors that bind the protein [108]. Therefore, evaluating the effect of incorporating residue physico-chemical properties and other information into the description of structural fragments on the performance of the structural footprinting method is an interesting direction for future work.

7.2 Dynamic Formation of Multiprotein Complexes

Mature experimental techniques exist that allow the inference of protein interactions, and recent proteomic studies used these and other technologies to characterize protein interactions among the components of many cellular processes. Even though the protein interaction networks for many cellular processes are available, we have little knowledge of the dynamical properties of protein interactions involved in these processes.

In Chapter 5 we proposed a tree-like representation, called a Tree of Complexes representation, of cellular processes. In our representation the nodes correspond to pseudo-complexes formed during the process. Moreover, the representation satisfies the additional condition that pseudo-complexes that contain any given protein induce a connected subgraph of the underlying tree. In this way, the representation captures not only the overlap

between the pseudo-complexes but also the manner in which proteins enter and leave their enclosing pseudo-complexes. If the formation of pseudo-complexes during the process follows a specific order, our representation can be used to hypothesize about this order. Indeed, once the root of the tree is fixed, the representation induces a partial order between the pseudo-complexes in the module, which in turn can be used to infer temporal relationships.

We relied on structural and algorithmic results for two well-studied graph families, chordal graphs and cographs, to develop an automatic method that extracts pseudo-complexes from a protein interaction network underlying a cellular process and outputs all valid Tree of Complexes representations of the process. Even though a Tree of Complexes representation is not unique, the protein interaction networks that we analyzed admit very few alternative tree topologies. For example, the pheromone signalling pathway admits twelve and the replication module three very closely related Tree of Complexes representations.

Our method generalizes the previous approach, due to Farach-Colton and colleagues [34], of extracting temporal information from the topology of a protein interaction network. As opposed to this previous approach, our method accounts for two phenomena clearly illustrated in the pheromone signaling pathway described in Section 5.3.1. First, the dynamic complex formation does not always follow a linear pathway but rather has a tree structure, where various branches correspond to the activation of different response systems. Our method allows us to model such processes by utilizing chordal graphs and their corresponding clique trees, rather than interval graphs, to model the complex formation. Second, many multi-protein complexes have several variants. For example, the MAPK complex centered at the scaffold protein `STE5` includes either `KSS1` or `FUS3`, but not

both. Our method explicitly accommodates these situations through pseudo-complexes which are modeled with cographs and their corresponding modular decomposition. It should be noted that cographs and their modular decomposition were previously used by Gagneur *et al.* to expose the hierarchical organization of protein complexes [43].

Although our algorithm is not guaranteed to produce a Tree of Complexes representation for every possible protein interaction network, the algorithm will succeed for a broad family of graphs, which includes chordal graphs (and thus interval graphs) and cographs. Currently, our method can be applied to protein interaction networks that do not contain long (longer than four nodes) chordless cycles. We distinguish between two different types of problematic networks for our method. The first type includes networks for which imposing a temporal order that encompasses all pseudo-complexes in the network is meaningless. The second type includes networks for which such order is meaningful, but the assumption that the complex formation has a tree-like structure is not valid. It would be interesting to investigate the extension of our approach to deal with networks of the second type by utilizing graph-theoretical tools developed for other specialized graph families, such as circular-arc graphs.

7.3 Topological Determinants of Lethality

The enrichment of high-degree nodes in essential proteins, known as the *centrality-lethality rule*, suggested that the topological prominence of a protein in a protein interaction network may be a good predictor of its biological importance. There exist numerous measures of topological prominence or *network centrality indices*; local centrality indices assign centrality values based on the topology of the node's local neighborhood whereas betweenness centrality indices assign centrality values based on the node's role in main-

taining the connectivity between pairs of other nodes in the network. Even though by definition degree centrality is a local measure, depending on the structure of the network hubs may play an important role in maintaining the overall connectivity of the network. In this thesis we sought to identify the main topological determinant of essentiality and a biological explanation for the connection between the network topology and essentiality.

To address this question in Chapter 6 we performed a rigorous analysis of five protein interaction networks for *Saccharomyces cerevisiae* compiled from diverse sources of interaction evidence. To clarify the topological role of essential proteins in general and essential hubs in particular we compared degree centrality to other local and betweenness centrality indices. We found that while in some networks high-degree nodes are as important in maintaining the overall network connectivity as nodes having high betweenness centrality values, this property is not due to essential proteins. On the contrary, essential proteins are indistinguishable in that respect from non-essential proteins having the same degree distribution. We also found that degree centrality is a better predictor of essentiality than any other measure tested and that correlation of betweenness indices with essentiality is entirely due to their correlation with degree centrality. Thus, we conclude that the topological determinant of essentiality is the node's local neighborhood rather than its role in maintaining the overall connectivity of the network.

Next we examined whether the *essential interactions model*, recently proposed to explain the centrality-lethality rule, is valid in the tested networks. We found that the model's central assumption that the majority of proteins are essential due to their involvement in one or more essential protein interactions, which are distributed uniformly at random among the edges of the network, violates basic clustering patterns of essential proteins in the networks we examined. The uniform distribution of essential protein inter-

actions implies that, as long as two proteins do not interact, the essentiality of one protein in the pair is independent of the essentiality of the other protein. However, in real protein interaction networks the essentiality of pairs of proteins that share many neighbors is correlated and the number of non-adjacent protein pairs that share three or more neighbors and are either both essential or both non-essential significantly deviates from the expected number of such pairs under the model. Consequently, we rejected the essential interactions explanation with high-confidence.

The above observations led us to propose an alternative explanation for the centrality-lethality rule in terms of ECOBIMs which are biological processes that are: (i) essential for an organism's vitality as measured by the large fraction of essential proteins and (ii) composed of proteins that interact extensively with each other. We developed two complementary methods to extract putative ECOBIMs from a protein interaction network. Both methods rely on GO biological process annotation and look for densely connected, essential subnetworks of proteins that are annotated with the same GO biological term.

We demonstrated that the membership in ECOBIMs accounts for the centrality-lethality rule in the tested networks. In particular we showed that the majority of essential hubs belong to one or more ECOBIMs, and hubs that are not members of ECOBIMs are depleted in essential proteins. Furthermore, for proteins that are not members of ECOBIMs the correlation of degree centrality with essentiality is due to the number of ECOBIMs neighbors; i.e., high-degree nodes that have few neighbors in ECOBIMs are not enriched in essential proteins.

The outcome of our experiments results point to an interesting direction for future investigation. We demonstrated that there are no grounds for a causal relationship between the topological prominence, as measured by betweenness centrality, and essentiality. Does

this observation hold if we consider other definitions of biological importance? While essential proteins are not distinguishable from non-essential proteins in their ability to disrupt the network, perhaps other biologically important proteins are. As pointed out in the study by Batada *et al.* [9], other indications of biological importance include the rate of evolution, tight control of the abundance and activity as measured by mRNA half-lives, and the number of phosphorylation sites.

Bibliography

[1] R. Albert, H. Jeong, and A. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, Jul 2000. doi: 10.1038/35019019. URL `http://dx.doi.org/10.1038/35019019`.

[2] V. Alesker, R. Nussinov, and H. Wolfson. Detection of non-topological motifs in protein structures. *Protein Engineering*, 9:1103–1119, 1996.

[3] P. Aloy and R. Russell. Potential artefacts in protein-interaction networks. *FEBS Letters*, 530(1-3):253–254, Oct 2002.

[4] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[5] S. Altschul, T. Madden, A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25:3389–3402, 1997.

[6] V. Archambault, E. Chang, B. Drapkin, F. Cross, B. Chait, and M. Rout. Targeted proteomic study of the cyclin-cdk module. *Molecular Cell*, 14:699–711, 2004. doi: 10.1016/j.molcel.2004.05.025. URL `http://dx.doi.org/10.1016/j.molcel.2004.05.025`.

[7] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature Genetics*, 25(1):25–29, May 2000. doi: 10.1038/75556. URL `http://dx.doi.org/10.1038/75556`.

[8] L. Bardwell. A walk-through of the yeast mating pheromone response pathway. *Peptides*, 26:339–350, 2005.

[9] N. Batada, L. Hurst, and M. Tyers. Evolutionary and physiological importance of hub proteins. *PLoS Computational Biology*, 2(7):e88, Jul 2006. doi: 10.1371/journal.pcbi.0020088. URL `http://dx.doi.org/10.1371/journal.pcbi.0020088`.

[10] N. Batada, T. Reguly, A. Breitkreutz, L. Boucher, B. Breitkreutz, L. Hurst, and M. Tyers. Stratus not altocumulus: a new view of the yeast protein interaction network. *PLoS Biology*, 4(10):e317, Sep 2006. doi: 10.1371/journal.pbio.0040317. URL `http://dx.doi.org/10.1371/journal.pbio.0040317`.

[11] V. Batagelj and A. Mrvar. Pajek - Program for large network analysis. *Connections*, 2:47–57, 1998.

[12] S. Bell and A. Dutta. DNA replication in eukaryotic cells. *Annu. Rev. Biochem.*, 71:333–374, 2002. doi: 10.1146/annurev.biochem.71.110601.135425. URL `http://dx.doi.org/10.1146/annurev.biochem.71.110601.135425`.

[13] S. Bell and A. Dutta. DNA replication in eukaryotic cells. *Annual Reviews in Biochemistry*, 71:333–374, 2002.

[14] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28: 235–242, 2000.

[15] P. Bernstein and N. Goodman. Power of natural semijoins. *SIAM Journal on Computing*, 10:751–771, 1981.

[16] J. Blair and B. Peyton. An introduction to chordal graphs and clique trees. In A. George, J. Gilbert, and J. Liu, editors, *Graph theory and sparse matrix computations*, pages 1–29. Springer, 1993.

[17] J. Bloom and C. Adami. Apparent dependence of protein evolutionary rate on number of interactions is linked to biases in protein-protein interactions data sets. *BMC Evolutionary Biology*, 3:21, Oct 2003. doi: 10.1186/1471-2148-3-21. URL `http://dx.doi.org/10.1186/1471-2148-3-21`.

[18] P. Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2(1):113–120, 1972.

[19] T. Bouwmeester, A. Bauch, H. Ruffner, P. Angrand, G. Bergamini, K. Croughton, C. Cruciat, D. Eberhard, J. Gagneur, and S. Ghidelli. A physical and functional map of the human TNF-alpha/NF-kappaB signal transduction pathway. *Nature Cell Biology*, 6:97–105, 2004. doi: http://dx.doi.org/10.1038/ncb1086.

[20] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[21] J. E. Bray, A. E. Todd, F. M. Pearl, J. M. Thornton, and C. A. Orengo. The CATH Dictionary of Homologous Superfamilies (DHS): a consensus approach for identifying distant structural homologues. *Protein Engineering*, 13(3):153–165, Mar 2000.

[22] O. Carugo and S. Pongor. Protein fold similarity estimated by a probabilistic ap-

proach based on c[alpha]-c[alpha] distance comparison. *Journal of Molecular Biology*, 315:887–898, 2002.

[23] I. Choi, J. Kwon, and S. Kim. Local feature frequency profile: a method to measure structural similarity in proteins. *Proceedings of the National Academy of Sciences of USA*, 101:3797–3802, 2004.

[24] C. Chothia and A. M. Lesk. The relation between the divergence of sequence and structure in proteins. *The EMBO Journal*, 5(4):823–826, Apr 1986.

[25] S. Collins, P. Kemmeren, X. Zhao, J. Greenblatt, F. Spencer, F. Holstege, J. Weissman, and N. Krogan. Toward a comprehensive atlas of the physical interactome of Saccharomyces cerevisiae. *Molecular and Cellular Proteomics*, 6(3):439–450, Mar 2007. doi: 10.1074/mcp.M600381-MCP200. URL `http://dx.doi.org/10.1074/mcp.M600381-MCP200`.

[26] D. Corneil, Y. Perl, and L. Stewart. Complement reducible graphs. *Discrete Applied Mathematics*, 3:163–174, 1981.

[27] S. Coulomb, M. Bauer, D. Bernard, and M.-C. Marsolier-Kergoat. Gene essentiality and the topology of protein interaction networks. *Proceedings of the Royal Society B*, 272(1573):1721–1725, Aug 2005. doi: 10.1098/rspb.2005.3128. URL `http://dx.doi.org/10.1098/rspb.2005.3128`.

[28] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[29] C. Deane, L. Salwinski, I. Xenarios, and D. Eisenberg. Protein interactions: two methods for assessment of the reliability of high throughput observations. *Molecular and Cellular Proteomics*, 1(5):349–356, May 2002. doi: 10.1074/mcp.M100037-MCP200.

[30] W. DeLano. The PyMOL Molecular Graphics System, 2002. URL `http://www.pymol.org`.

[31] R. F. Doolittle. Searching through sequence databases. *Methods in Enzymology*, 183:99–110, 1990.

[32] R. Downey and M. Fellows. *Parameterized complexity*, pages 29–30. Springer, 1999.

[33] E. Estrada and J. Rodríguez-Velázquez. Subgraph centrality in complex networks. *Physical Review E*, 71(5):56103, 2005.

[34] M. Farach-Colton, Y. Huang, and J. Woolford. Discovering temporal relations in molecular pathways using protein-protein interactions. In *Proceedings of the 8th Annual International Conference on Research in Computational Molecular Biology*, pages 150–156, San Diego, California, USA, 2004. doi: 10.1145/974614.974635. URL http://doi.acm.org/10.1145/974614.974635.

[35] A. Fatica and D. Tollervey. Making ribosomes. *Curr. Opin. Cell Biol.*, 14(3):313–318, Jun 2002.

[36] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45 (4):634–652, 1998.

[37] R. Fenn. *Geometry*. Springer Undegraduate Mathematics Series. Springer-Verlag, 2001.

[38] S. Fields and O. Song. A novel genetic system to detect protein-protein interactions. *Nature*, 340:245–246, 1989. doi: http://dx.doi.org/10.1038/340245a0.

[39] L. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40 (1):35–41, 1977.

[40] M. Fromont-Racine, J. C. Rain, and P. Legrain. Toward a functional analysis of the yeast genome through exhaustive two-hybrid screens. *Nature Genetics*, 16(3):277–282, Jul 1997. doi: 10.1038/ng0797-277. URL http://dx.doi.org/10.1038/ng0797-277.

[41] M. Fromont-Racine, A. E. Mayes, A. Brunet-Simon, J. C. Rain, A. Colley, I. Dix, L. Decourty, N. Joly, F. Ricard, J. D. Beggs, and P. Legrain. Genome-wide protein interaction screens reveal functional networks involving Sm-like proteins. *Yeast*, 17 (2):95–110, Jun 2000. doi: 3.0.CO;2-H. URL http://dx.doi.org/3.0.CO;2-H.

[42] M. Fromont-Racine, B. Senger, C. Saveanu, and F. Fasiolo. Ribosome assembly in eukaryotes. *Gene*, 313:17–42, Aug 2003.

[43] J. Gagneur, R. Krause, T. Bouwmeester, and G. Casari. Modular decomposition of protein-protein interaction networks. *Genome Biology*, 5(8):R57, 2004. ISSN 1465-6906. doi: http://dx.doi.org/10.1186/gb-2004-5-8-r57. URL http://genomebiology.com/2004/5/8/R57.

[44] Z. Gaspari, K. Vlahovicek, and S. Pongor. Efficient recognition of folds in protein 3D structures by the improved PRIDE algorithm. *Bioinformatics*, 21(15):3322–3323, Aug 2005. doi: 10.1093/bioinformatics/bti513. URL http://dx.doi.org/10.1093/bioinformatics/bti513.

[45] A. Gavin, M. Bosche, R. Krause, P. Grandi, M. Marzioch, A. Bauer, J. Schultz, J. Rick, A. Michon, and C. Cruciat. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415:141–147, 2002. doi: http://dx.doi.org/10.1038/415141a.

[46] A.-C. Gavin, P. Aloy, P. Grandi, R. Krause, M. Boesche, M. Marzioch, C. Rau, L. J. Jensen, S. Bastuck, B. Dmpelfeld, A. Edelmann, M.-A. Heurtier, V. Hoffman, C. Hoefert, K. Klein, M. Hudak, A.-M. Michon, M. Schelder, M. Schirle, M. Remor, T. Rudi, S. Hooper, A. Bauer, T. Bouwmeester, G. Casari, G. Drewes, G. Neubauer, J. M. Rick, B. Kuster, P. Bork, R. B. Russell, and G. Superti-Furga. Proteome survey reveals modularity of the yeast cell machinery. *Nature*, 440(7084):631–636, Mar 2006. doi: 10.1038/nature04532. URL `http://dx.doi.org/10.1038/nature04532`.

[47] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory (B)*, 16:47–56, 1974.

[48] M. Gerstein and M. Levitt. Comprehensive assessment of automatic structural alignment against a manual standard, the SCOP classification of proteins. *Protein Science*, 7:445–456, 1998.

[49] G. Giaever, A. M. Chu, L. Ni, C. Connelly, L. Riles, S. Vronneau, S. Dow, A. Lucau-Danila, K. Anderson, B. Andr, A. P. Arkin, A. Astromoff, M. El-Bakkoury, R. Bangham, R. Benito, S. Brachat, S. Campanaro, M. Curtiss, K. Davis, A. Deutschbauer, K.-D. Entian, P. Flaherty, F. Foury, D. J. Garfinkel, M. Gerstein, D. Gotte, U. Gldener, J. H. Hegemann, S. Hempel, Z. Herman, D. F. Jaramillo, D. E. Kelly, S. L. Kelly, P. Ktter, D. LaBonte, D. C. Lamb, N. Lan, H. Liang, H. Liao, L. Liu, C. Luo, M. Lussier, R. Mao, P. Menard, S. L. Ooi, J. L. Revuelta, C. J. Roberts, M. Rose, P. Ross-Macdonald, B. Scherens, G. Schimmack, B. Shafer, D. D. Shoemaker, S. Sookhai-Mahadeo, R. K. Storms, J. N. Strathern, G. Valle, M. Voet, G. Volckaert, C. yun Wang, T. R. Ward, J. Wilhelmy, E. A. Winzeler, Y. Yang, G. Yen, E. Youngman, K. Yu, H. Bussey, J. D. Boeke, M. Snyder, P. Philippsen, R. W. Davis, and M. Johnston. Functional profiling of the Saccharomyces cerevisiae genome. *Nature*, 418(6896):387–391, Jul 2002. doi: 10.1038/nature00935. URL `http://dx.doi.org/10.1038/nature00935`.

[50] J. Gibrat, T. Madej, and S. Bryant. Surprising similarities in structure comparison. *Current Opinion in Structural Biology*, 6:377–385, 1996.

[51] L. Giot, J. S. Bader, C. Brouwer, A. Chaudhuri, B. Kuang, Y. Li, Y. L. Hao, C. E. Ooi, B. Godwin, E. Vitols, G. Vijayadamodar, P. Pochart, H. Machineni, M. Welsh, Y. Kong, B. Zerhusen, R. Malcolm, Z. Varrone, A. Collis, M. Minto, S. Burgess, L. McDaniel, E. Stimpson, F. Spriggs, J. Williams, K. Neurath, N. Ioime, M. Agee, E. Voss, K. Furtak, R. Renzulli, N. Aanensen, S. Carrolla, E. Bickelhaupt, Y. Lazovatsky, A. DaSilva, J. Zhong, C. A. Stanyon, R. L. Finley, K. P. White, M. Braverman, T. Jarvie, S. Gold, M. Leach, J. Knight, R. A. Shimkets,

M. P. McKenna, J. Chant, and J. M. Rothberg. A protein interaction map of Drosophila melanogaster. *Science*, 302(5651):1727–1736, Dec 2003. doi: 10.1126/science.1090289. URL `http://dx.doi.org/10.1126/science.1090289`.

[52] A. Goldberg and S. Rao. Flows in undirected unit capacity networks. *SIAM Journal of Discrete Mathematics*, 12(1):1–5, 1999.

[53] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier, second edition, 2004.

[54] R. Gomory and T. Hu. Multi-terminal network flows. *SIAM Journal of Applied Mathematics*, 9:551–570, 1961.

[55] D. S. Goodsell. *The machinery of life*. Springer, 1997.

[56] M. Gribskov and N. L. Robinson. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Computers and Chemistry*, 20(1):25–33, Mar 1996.

[57] D. Gusfield. Very simple methods for all pairs network flow analysis. *SIAM Journal on Computing*, 19:143–155, 1990.

[58] M. Hahn and A. Kern. Comparative genomics of centrality and essentiality in three eukaryotic protein-interaction networks. *Molecular Biology and Evolution*, 22(4):803–806, Apr 2005. doi: 10.1093/molbev/msi072. URL `http://dx.doi.org/10.1093/molbev/msi072`.

[59] M. Hampsey. Molecular genetics of the RNA polymerase II general transcriptional machinery. *Microbiology and Molecular Biology Reviews*, 62(2):465–503, Jun 1998.

[60] G. T. Hart, I. Lee, and E. Marcotte. A high-accuracy consensus map of yeast protein complexes reveals modular nature of gene essentiality. *BMC Bioinformatics*, 8(1):236, Jul 2007. doi: 10.1186/1471-2105-8-236. URL `http://dx.doi.org/10.1186/1471-2105-8-236`.

[61] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature*, 402:C47–C52, 1999. doi: http://dx.doi.org/10.1038/35011540.

[62] X. He and J. Zhang. Why do hubs tend to be essential in protein networks? *PLoS Genetics*, 2(6):e88, Jun 2006. doi: 10.1371/journal.pgen.0020088. URL `http://dx.doi.org/10.1371/journal.pgen.0020088`.

[63] C. Ho and R. Lee. Counting clique trees and computing perfect elimination schemes in parallel. *Information Processing Letters*, 31:61–68, 1989.

[64] Y. Ho, A. Gruhler, A. Heilbut, G. Bader, L. Moore, S. Adams, A. Millar, P. Taylor, K. Bennett, and K. Boutilier. Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry. *Nature*, 415:180–183, 2002. doi: http://dx.doi.org/10.1038/415180a.

[65] L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233:123–138, 1993.

[66] L. Holm and C. Sander. 3-D Lookup: fast protein structure database searches at 90% reliability. In *Proceedings of Intelligent Systems in Molecular Biology*, 1995.

[67] L. Holm and C. Sander. Mapping the protein universe. *Science*, 273(5275):595–603, Aug 1996.

[68] J. Hou, S. Jun, C. Zhang, and S. Kim. Global mapping of the protein structure space and application in structure-based inference of protein function. *Proceedings of the National Academy of Sciences of USA*, 102(10):3651–3656, Mar 2005. doi: 10.1073/pnas.0409772102. URL `http://dx.doi.org/10.1073/pnas.0409772102`.

[69] T. Ito, K. Tashiro, S. Muta, R. Ozawa, T. Chiba, M. Nishizawa, K. Yamamoto, S. Kuhara, and Y. Sakaki. Toward a protein-protein interaction map of the budding yeast: A comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins. *Proceedings of the National Academy of Sciences of USA*, 97(3):1143–1147, Feb 2000.

[70] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proceedings of the National Academy of Sciences of USA*, 98:4569–4574, 2001. doi: http://dx.doi.org/10.1073/pnas.061034498.

[71] A. Jain and R. Dubes. *Algorithms for clustering data*. Prentice Hall, 1988.

[72] R. Jansen, H. Yu, D. Greenbaum, Y. Kluger, N. Krogan, S. Chung, A. Emili, M. Snyder, J. Greenblatt, and M. Gerstein. A Bayesian Networks approach for predicting protein-protein interactions from genomic data. *Science*, 302(5644):449–453, 2003. URL `http://www.sciencemag.org/cgi/content/abstract/302/5644/449`.

[73] H. Jeong, S. P. Mason, A. L. Barabasi, and Z. N. Oltvai. Lethality and centrality in protein networks. *Science*, 411:41–42, 2001.

[74] T. Joachims. Making large-scale Support Vector Machine learning practical. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector learning*, pages 169–184. The MIT Press, 1998.

[75] W. Kabsch and C. Sander. Secondary structure definition by the program DSSP. *Biopolymers*, 22:2577–2637, 1983.

[76] T. Kawabata and K. Nishikawa. Protein structure comparison using the Markov transition model of evolution. *Proteins*, 41:108–122, 2000.

[77] S. Kerrien, Y. Alam-Faruque, B. Aranda, I. Bancarz, A. Bridge, C. Derow, E. Dimmer, M. Feuermann, A. Friedrichsen, R. Huntley, C. Kohler, J. Khadake, C. Leroy, A. Liban, C. Lieftink, L. Montecchi-Palazzi, S. Orchard, J. Risse, K. Robbe, B. Roechert, D. Thorneycroft, Y. Zhang, R. Apweiler, and H. Hermjakob. IntAct-open source resource for molecular interaction data. *Nucleic Acids Research*, 35(Database issue):D561–D565, Jan 2007. doi: 10.1093/nar/gkl958. URL `http://dx.doi.org/10.1093/nar/gkl958`.

[78] R. Kolodny and N. Linial. Approximate protein structural alignment in polynomial time. *Proceedings of the National Academy of Sciences of USA*, 101:12201–12206, 2004.

[79] D. Koschützki, K. Lehmann, L. Peeters, S. Richter, D. Tenfelde-Podehl, and O. Zlotowski. Centrality indices. In U. Brandes and T. Erlebach, editors, *Network analysis: methodological foundations*, pages 16–61. Springer, 2005.

[80] N. J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis, T. Punna, J. M. Peregrn-Alvarez, M. Shales, X. Zhang, M. Davey, M. D. Robinson, A. Paccanaro, J. E. Bray, A. Sheung, B. Beattie, D. P. Richards, V. Canadien, A. Lalev, F. Mena, P. Wong, A. Starostine, M. M. Canete, J. Vlasblom, S. Wu, C. Orsi, S. R. Collins, S. Chandran, R. Haw, J. J. Rilstone, K. Gandi, N. J. Thompson, G. Musso, P. S. Onge, S. Ghanny, M. H. Y. Lam, G. Butland, A. M. Altaf-Ul, S. Kanaya, A. Shilatifard, E. O'Shea, J. S. Weissman, C. J. Ingles, T. R. Hughes, J. Parkinson, M. Gerstein, S. J. Wodak, A. Emili, and J. F. Greenblatt. Global landscape of protein complexes in the yeast Saccharomyces cerevisiae. *Nature*, 440(7084):637–643, Mar 2006. doi: 10.1038/nature04670. URL `http://dx.doi.org/10.1038/nature04670`.

[81] P. Legrain, J. Wojcik, and J. M. Gauthier. Protein-protein interaction maps: a lead towards cellular functions. *Trends in Genetics*, 17(6):346–352, Jun 2001.

[82] S. Li, C. M. Armstrong, N. Bertin, H. Ge, S. Milstein, M. Boxem, P.-O. Vidalain, J.-D. J. Han, A. Chesneau, T. Hao, D. S. Goldberg, N. Li, M. Martinez, J.-F. Rual, P. Lamesch, L. Xu, M. Tewari, S. L. Wong, L. V. Zhang, G. F. Berriz, L. Jaco-

tot, P. Vaglio, J. Reboul, T. Hirozane-Kishikawa, Q. Li, H. W. Gabel, A. Elewa, B. Baumgartner, D. J. Rose, H. Yu, S. Bosak, R. Sequerra, A. Fraser, S. E. Mango, W. M. Saxton, S. Strome, S. V. D. Heuvel, F. Piano, J. Vandenhaute, C. Sardet, M. Gerstein, L. Doucette-Stamm, K. C. Gunsalus, J. W. Harper, M. E. Cusick, F. P. Roth, D. E. Hill, and M. Vidal. A map of the interactome network of the metazoan C. elegans. *Science*, 303(5657):540–543, Jan 2004. doi: 10.1126/science.1091403. URL `http://dx.doi.org/10.1126/science.1091403`.

[83] H. Lodhi, G. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

[84] A. Martin. The ups and downs of protein topology; rapid comparison of protein structure. *Protein Engineering*, 13:829–837, 2000.

[85] T. A. McKee and F. McMorris. *Topics in intersection graph theory*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 1999.

[86] K. Menger. Zur allgemeinen kurventhoerie. *Fund. Math.*, 10:96–115, 1927.

[87] J. Mestre, 2007. Personal communication.

[88] H. W. Mewes, D. Frishman, K. F. X. Mayer, M. Münsterkötter, O. Noubibou, P. Pagel, T. Rattei, M. Oesterheld, A. Ruepp, and V. Stümpflen. MIPS: analysis and annotation of proteins from whole genomes in 2005. *Nucleic Acids Research*, 34(Database issue):D169–D172, Jan 2006. doi: 10.1093/nar/gkj148. URL `http://dx.doi.org/10.1093/nar/gkj148`.

[89] R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, 2004. URL `http://www.sciencemag.org/cgi/content/abstract/303/5663/1538`.

[90] R. Mrowka, A. Patzak, and H. Herzel. Is there a bias in proteome research? *Genome Research*, 11(12):1971–1973, Dec 2001. doi: 10.1101/gr.206701. URL `http://dx.doi.org/10.1101/gr.206701`.

[91] A. Murzin, S. Brenner, T. Hubbard, and C. Chotia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.

[92] C. Myers, D. Barrett, M. Hibbs, C. Huttenhower, and O. Troyanskaya. Finding function: evaluation methods for functional genomic data.

BMC Genomics, 7:187, 2006. doi: 10.1186/1471-2164-7-187. URL http://dx.doi.org/10.1186/1471-2164-7-187.

[93] M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27:39–54, 2003. URL http://arxiv.org/abs/cond-mat/0309045v1.

[94] R. Nussinov and H. Wolfson. Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques. *Proceedings of the National Academy of Sciences of USA*, 88:10495–10499, 1991.

[95] C. Orengo, N. Brown, and W. Taylor. Fast structure alignment for protein databank searching. *Proteins*, 14:139–167, 1992.

[96] C. Orengo, A. Michie, S. Jones, D. Jones, M. Swindells, and J. Thornton. CATH - A hierarchic classification of protein domain structures. *Structure*, 5:1093–1108, 1997.

[97] B. Parlett. *The symmetric eigenvalue problem*. Prentice-Hall, 1980.

[98] F. Pearl, D. Lee, J. Bray, I. Sillitoe, A. Todd, A. Harrison, J. Thornton, and C. Orengo. Assigning genomic sequences to CATH. *Nucleic Acids Research*, 28: 277–282, 2000.

[99] W. Pearson. Searching protein sequence libraries: comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. *Genomics*, 11:635–650, 1991.

[100] O. Redfern, G. Alastair, M. Maibaum, and C. Orengo. Survey of current protein family databases and their application in comparative, structural and functional genomics. *Journal of Chromatography B*, 815:97–107, 2005.

[101] T. Reguly, A. Breitkreutz, L. Boucher, B.-J. Breitkreutz, G. C. Hon, C. L. Myers, A. Parsons, H. Friesen, R. Oughtred, A. Tong, C. Stark, Y. Ho, D. Botstein, B. Andrews, C. Boone, O. Troyanskya, T. Ideker, K. Dolinski, N. Batada, and M. Tyers. Comprehensive curation and analysis of global interaction networks in Saccharomyces cerevisiae. *Journal of Biology*, 5(4):11, 2006. doi: 10.1186/jbiol36. URL http://dx.doi.org/10.1186/jbiol36.

[102] G. Rigaut, A. Shevchenko, B. Rutz, M. Wilm, M. Mann, and B. Seraphin. A generic protein purification method for protein complex characterization and proteome exploration. *Nat. Biotechnol.*, 17:1030–1032, 1999. doi: http://dx.doi.org/10.1038/13732.

[103] P. Rogen and H. Bohr. A new family of protein shape descriptors. *Mathematical Biosciences*, 182:167–181, 2003.

[104] P. Rogen and B. Fain. Automatic classification of protein structure by using Gauss integrals. *Proceedings of the National Academy of Sciences of USA*, 100:119–124, 2003.

[105] B. Rost. Protein structures sustain evolutionary drift. *Folding and Design*, 2(3): S19–S24, 1997.

[106] J.-F. Rual, K. Venkatesan, T. Hao, T. Hirozane-Kishikawa, A. Dricot, N. Li, G. F. Berriz, F. D. Gibbons, M. Dreze, N. Ayivi-Guedehoussou, N. Klitgord, C. Simon, M. Boxem, S. Milstein, J. Rosenberg, D. S. Goldberg, L. V. Zhang, S. L. Wong, G. Franklin, S. Li, J. S. Albala, J. Lim, C. Fraughton, E. Llamosas, S. Cevik, C. Bex, P. Lamesch, R. S. Sikorski, J. Vandenhaute, H. Y. Zoghbi, A. Smolyar, S. Bosak, R. Sequerra, L. Doucette-Stamm, M. E. Cusick, D. E. Hill, F. P. Roth, and M. Vidal. Towards a proteome-scale map of the human protein-protein interaction network. *Nature*, 437(7062):1173–1178, Oct 2005. doi: 10.1038/nature04209. URL `http://dx.doi.org/10.1038/nature04209`.

[107] L. Salwinski, C. Miller, A. Smith, F. Pettit, J. Bowie, and D. Eisenberg. The Database of Interacting Proteins: the 2004 update. *Nucleic Acids Research*, 32: D449–D451, 2004.

[108] O. Sander, T. Sing, I. Sommer, A. Low, P. Cheung, P. R. Harrigan, T. Lengauer, and F. Domingues. Structural descriptors of gp120 V3 loop for the prediction of HIV-1 coreceptor usage. *PLoS Computational Biology*, 3(3):e58, Mar 2007. doi: 10.1371/journal.pcbi.0030058. URL `http://dx.doi.org/10.1371/journal.pcbi.0030058`.

[109] J. M. Sauder, J. W. Arthur, and R. L. Dunbrack. Large-scale comparison of protein sequence alignment algorithms with structure alignments. *Proteins*, 40(1):6–22, Jul 2000.

[110] D. Sheskin. *Handbook of parametric and nonparametric statistical procedures*. Chapman and Hall CRC, fourth edition, 2007.

[111] I. Shindyalov and P. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Engineering*, 11:739–747, 1998.

[112] B. Shoemaker and A. Panchenko. Deciphering protein-protein interactions. Part I. Experimental techniques and databases. *PLoS Computational Biology*, 3(3):e42, Mar 2007. doi: 10.1371/journal.pcbi.0030042. URL `http://dx.doi.org/10.1371/journal.pcbi.0030042`.

[113] M. Sierk and W. Pearson. Sensitivity and selectivity in protein structure comparison. *Protein Science*, 13:773–785, 2004.

[114] P. Slavík. Improved performance of the greedy algorithm for partial cover. *Information Processing Letters*, 64(5):251–254, 1997.

[115] E. Sprinzak, S. Sattath, and H. Margalit. How reliable are experimental protein-protein interaction data? *Journal of Molecular Biology*, 327(5):919–923, Apr 2003. doi: 10.1016/S0022-2836(03)00239-0.

[116] C. Stark, B. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers. BioGRID: a general repository for interaction datasets. *Nucleic Acids Research*, 34(Database issue):D535–D539, Jan 2006. doi: 10.1093/nar/gkj109. URL `http://dx.doi.org/10.1093/nar/gkj109`.

[117] G. Strang. *Linear algebra and its applications*. Brooks Cole, 3rd edition, 1988.

[118] P. Uetz, L. Giot, G. Cagney, T. Mansfield, R. Judson, J. Knight, D. Lockshon, V. Narayan, M. Srinivasan, and P. Pochart. A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. *Nature*, 403:623–627, 2000. doi: 10.1038/35001009.

[119] C. von Mering, R. Krause, B. Snel, M. Cornell, S. Oliver, S. Fields, and P. Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417(6887):399–403, May 2002. doi: 10.1038/nature750. URL `http://dx.doi.org/10.1038/nature750`.

[120] H. Yu, D. Greenbaum, H. X. Lu, X. Zhu, and M. Gerstein. Genomic analysis of essentiality within protein networks. *Trends in Genetics*, 20(6):227–231, Jun 2004. doi: 10.1016/j.tig.2004.04.008. URL `http://dx.doi.org/10.1016/j.tig.2004.04.008`.

[121] H. Yu, P. M. Kim, E. Sprecher, V. Trifonov, and M. Gerstein. The importance of bottlenecks in protein networks: Correlation with gene essentiality and expression dynamics. *PLoS Computational Biology*, 3(4):e59, Apr 2007. doi: 10.1371/journal.pcbi.0030059. URL `http://dx.doi.org/10.1371/journal.pcbi.0030059`.