Charles University

Faculty of Science

Study programme: Bioinformatics

Branch of study: Bioinformatics

Nikola Kalábová

Applications of graph theory in protein function prediction

Aplikace teorie grafů v predikci funkce proteinů

Bachelor's thesis

Supervisor: Ing. David Hartman, Ph.D.

Prague 2021

I would like to thank my supervisor Ing. David Hartman, Ph.D., for his guidance, providing insightful feedback, and making time for me whenever needed despite his busy schedule. I would also like to thank my family for supporting me during my studies. A special thanks to the authors of this LaTeXtemplate for the overall delicate appearance of this thesis.

Title: Applications of graph theory in protein function prediction

Author: Nikola Kalábová

Supervisor: Ing. David Hartman, Ph.D., Computer Science Institute of Charles University

Abstract: The rapid development of the whole-genome sequencing methods and their reducing cost resulted in a huge number of sequenced genomes. Developing reliable methods for in-silico annotation of the expeditiously growing number of sequenced genomes is the next challenge of modern biology. We described a graph-theoretical approach for function prediction from the protein-protein interaction networks and outlined its strengths and weaknesses. We illustrate the principles of this approach on selected algorithms based on different ideas and provide their comparison and evaluation.

Abstrakt: Rapidní vývoj celogenomových sekvenačních metod a jejich snižující se cena zapříčinila existenci velkého množství osekvenovaných genomů. Vývoj spolehlivých in-silico metod pro anotaci rychle rostoucího počtu osekvenovaných genomů představuje výzvu pro moderní biologii. V práci představujeme způsob predikce funkce proteinů, založený na aplikaci teorie grafů v protein-protein interakčních sítích a identifikujeme jeho silné a slabé stránky. Tento přístup poté ilustrujeme na vybraných algoritmech založených na různých myšlenkách. Představené algoritmy porovnáváme a vyhodnocujeme jejich spolehlivost.

Keywords: protein function prediction, graph algorithms, protein-protein interactions, protein-protein interaction networks, graph theory

Contents

List of Abbreviations

PPI Protein-protein interactions

PPIN Protein-protein interaction networks

GO Gene Onthology

MCL Markov Clustering Algorithm

TSP Traveling Salesman Problem

ACO-MAE Ant Colony Optimization with Multi-Agent Evolution

LLPA Link-driven Label Propagation Algorithm

Introduction

Proteins perform a vast number of tasks in organisms, from defining the structure of cells or transporting vital molecules to replicating DNA or catalyzing metabolical reactions. Therefore, determining the function of proteins is an essential task in modern biology and biochemistry. Understanding the function of single proteins helps us understand the complex processes in cells. Knowing which proteins are involved in which processes is essential for personalized medicine because it helps in addressing a specialized problem without disrupting other vital cell processes [82].

In recent years, rapid development in sequencing technologies was seen, resulting in many sequenced genomes [69, 45]. The next big challenge of modern biology is the annotation of the genomes and the related determination of the function of individual translated proteins [25] because even in the well-studied organisms, a large portion of the genome remains uncharacterized [86].

Traditionally protein function was studied for just a few proteins at a time [25], through methods using for example the clustering patterns of coregulated genes [79], sequence similarity [61], or phylogenetic profiles [71]. Every one of these methods has its own limitations.
Proteins rarely act as independent units. Knowing the function and the interactions of one protein usually helps understand the function of interacting proteins [86]. The availability of whole-genome sequences and the ability to detect coexpression patterns shifted the research focus from single proteins and complexes of few proteins to large complexes and even the entire proteome [91].
In recent years high-throughput experimental and in-silico techniques for protein-protein interactions, such as protein complex purification techniques using mass spectrometry, correlated messenger RNA expression profiles, genetic interaction data, gene fusion, gene neighborhood, and gene co-occurrences, have been developed [89, 75], which enabled the encoding of protein-protein interaction into complex networks.
Protein-protein interaction networks was first used for function prediction in Bartel et al. [8]. The prediction methods based on protein-protein interaction networks open wholly new possibilities. With the help of protein-protein interaction networks, we can now predict the function of all proteins with known interactions in a whole organism at once [40], even those proteins that suffer from low coverage of homologous or associated sequences, or similar structures [98].

This theses aims to present the graph-theoretical concept of function prediction from PPINs, outline its benefits and introduce in detail different widely used algorithms for function prediction from PPINs and compare them with each other.
In the first chapter, the biological background of protein annotation and protein function is covered, in the second chapter, different approaches for protein function prediction are discussed, and their shortcomings are marked. In the third chapter, the theoretical concepts of network-based methods are introduced, and

a short overview of available algorithms is given. In the fourth chapter, different categories of these algorithms are presented, with an example of one widely used algorithm from each category. The last chapter focuses on the evaluation and comparison of the presented algorithms.

1. Biological background

1.1 Genome sequencing

Due to the development of high throughput new generation sequencing methods, such as Illumina HiSeq [1] and the advancement of modern bioinformatics tools, the complexity and cost of genome sequencing are decreasing steadily [2]. This enables the sequenation of a large number of different genomes; hence the number of sequenced genomes is increasing rapidly. Because of that, the need for identifying relevant regions in genomes and their annotation arises.

1.2 Identifying protein coding regions

The task relevant for this work is the identification of the protein-coding regions. There are different approaches for their detection, presented for example in Lin et al. [57], Clamp et al. [15]. Thanks to these methods, we can predict the existence of proteins we never experimentally detected. Owing to the availability of a wide range of genomes, we are now in a situation where many protein sequences are predicted but never detected [12]. Thus, we have to resort to in-silico methods for their analysis. There is also a need for a precise definition of function and a human- and computer-readable database of annotated genes. This is accomplished in the Gene Ontology Consortium[3] (GO)

1.3 Gene ontology project

Currently, the GO includes experimental findings from over 150,000 published papers, represented as over 700,000 experimentally supported annotations.
GO differentiates three types of function annotations. The **molecular function**, which describes the molecular activities of individual gene products, **cellular component**, that is the location where molecular activities of individual gene products are active and **biological process**, which defines the pathways and more extensive processes to which that gene product's activity contributes.

1.3.1 Molecular function

Examples of molecular functions are adenylate cyclase activity or Toll-like receptor binding. The standard approach for molecular function determination is assigning the function according to sequentially or structurally similar proteins. For more accurate prediction of molecular function, binding sites are found through sequence-based methods such as Evolutionary Trace [56] or structure-based methods CASTp [20]. A local similarity search is then performed for the found binding

[1] Illumina Hiseq
[2] The Cost of Sequencing a Human Genome
[3] Gene Onthology

sites. ProFunc [50] combines global and local structure-based methods to predict molecular function.

1.3.2 Cellular component

Cellular components are, for example, mitochondria or ribosomes. Most newly synthesized proteins have a short signal peptide present at the N-terminus, which prompts a cell to translocate the protein. The respective DNA segment of the signal peptide is analyzed to determine the localization. An example of a tool used for protein localization is PSORT [66].

1.3.3 Biological process

The biological process is, for example, DNA repair or signal transduction. When predicting function from PPINs, this type of function is usually meant, though it is possible to this way also predict other types of function. In this work, the term protein function will refer to the biological process they are involved in.
Methods for predicting the biological process of proteins are explained in detail in the following chapters.

2. Methods for prediction of function and interactions

Function may be inferred from experimental methods [74, 64]. The same applies to protein-protein interactions [80, 42]. However, owing to the advances in genomics, with the number of detected proteins, it is impossible to examine them all experimentally. Moreover, these methods are not always applicable, are time and cost-consuming, and particularly problematic for transient complexes.

In recent years a considerable amount of in-silico methods emerged. The power of these methods is that they can predict function and interactions for proteins that have never been characterized. However, these methods also have their specific shortcomings.

2.1 Prediction of function and interactions

In this section, standard methods for predicting the function of proteins and interactions between them are presented. Notice that a protein in some way similar to the query protein with known function or interactions is required for every mentioned method.

2.1.1 Association-based methods

Domain fusion

The domain fusion concept was introduced in Marcotte et al. [63]. It is based on the idea that if two domains (spatially separated, evolutionary conserved, stable units of the protein structure) with two different polypeptides are found in the same polypeptide in another organism, these domains are with high probability functionally linked and are interacting with each other. The reason for that is that such two domains probably evolved from one single polypeptide, including both. This method suffers from low coverage, as it is not common to find two proteins fulfilling such conditions.

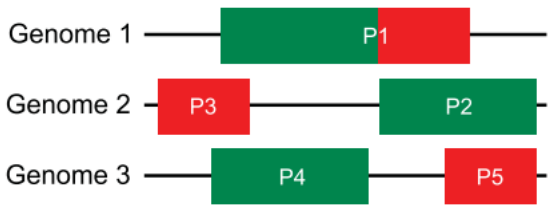


Figure 2.1: Proteins P2 and P3 in Genomes 2 and 3 are predicted to interact because their homologs are fused in the first genome. Source: Raman [79]

Phylogenetic profiles

This method is based on the idea that proteins with similar phylogenetic profiles (profiles describing the presence or absence of a protein in a set of reference genomes) are usually also linked functionally [71] because similar phylogenetic profiles produce similar phenotypes.

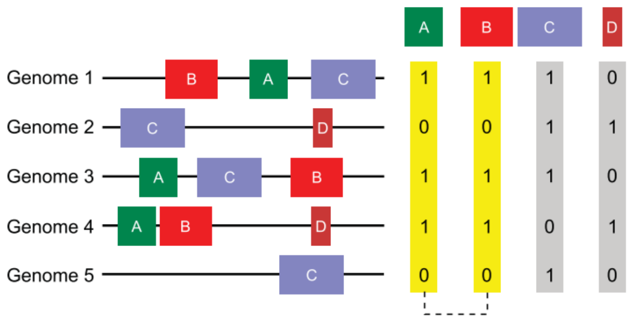


Figure 2.2: The figure shows five hypothetical genomes, containing the proteins A, B, C and D. The presence or absence of each protein is indicated by 1 or 0, respectively. Identical profiles are highlighted (dotted line). Source: Raman [79]

Co-evolution based methods

Co-evolution takes place when multiple species reciprocally affect each other's evolution through the process of natural selection. The species thus create mutual selective pressure on each other. This method was proposed in Barker and Pagel [7]. The main idea is that presence of several compensatory mutations in corresponding proteins can predict the coadaptation of the interacting proteins [79]. This approach can also be used to identify specific residues involved at the interaction sites [70].

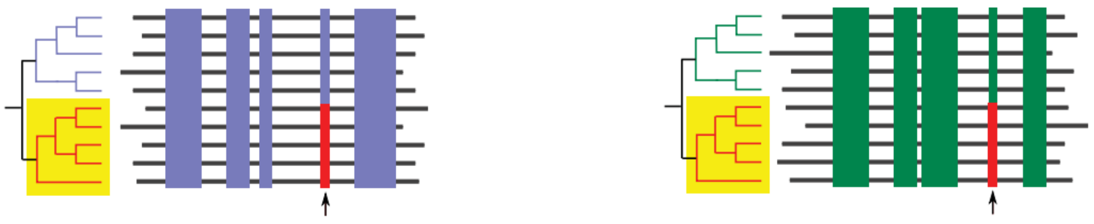


Figure 2.3: The alignments of two protein families are shown; conserved residues in either alignment are shown in the same colour (blue and green). Correlated mutations in either alignment (coloured red) are indicated by arrow marks. Source: Raman [79]

Problems

It may happen that even strong genomic association not only does not indicate function similarity but even indicate an opposite or complementary functional association [31].

2.1.2 Sequence based methods

Homologs with high sequence similarity tend to have similar functions. Homology-based methods are based on finding those homologs, for example, through BLAST Altschul et al. [3] or InterProScan [99], and assigning the same function to the query protein-coding gene. Recently other tools emerged, such as eggNOG-mapper [33].

For protein-protein interactions, the idea behind this approach is to two proteins from a query organism to find two respective homologs interacting with each other. High sequence similarity is required, especially in the binding site [5]. Moreover, we differentiate the ortholog-based approach and domain-based approach. The ortholog-based approach is based on finding two interacting orthologs (sequences found in different species that originated by vertical descent from a single gene of the last common ancestor) [52]. In contrast, the domain-based approach concentrates on finding domains that are known to interact with each other in inspected sequences because domains tend to be more evolutionary conserved, as well as their interactions [95], making this approach more reliable.

Problems

These methods are known to produce false positives [43]. Moreover, homologs with high sequence similarity and known function are required. More distant homologs can also offer functional clues but are less reliable [9]. The annotations also have to be reliable. This is often not the case, as many BLAST hits are hypothetical or electronically annotated proteins [31] and a possible error would be further propagated.
Another problem is that top hit sequences sometimes fail to represent the closest phylogenetic neighbor [31].

2.1.3 Structure based methods

In this work, we concentrate on proteins never experimentally measured. Therefore we first have to predict the structure. The reliability of structure prediction was recently hugely improved through AlphaFold [1], but it still brings additional errors to the prediction.
The structure may be predicted by homology modeling[24], ab-initio modeling [51] or threading methods [53], but all these have thir limitations [65].

For the prediction of the protein function is the concept similar to sequence-based methods for the prediction of PPIs. We are searching for two interacting proteins with respectively similar binding sites to our query proteins. These are then believed to interact with each other [100]. We search for those proteins that have compatible binding sites for function and interactions prediction.
Structure tends to be more conserved than sequence [94]; hence prediction from structural similarity tends to be more accurate. However, the prediction of struc-

[1]AlphaFold

ture is computationally demanding and results in additional inaccuracy.

2.2 Other methods for function prediction

2.2.1 Conserved neighbourhood

It was found that for those genes that encode proteins and are neighbors on a chromosome in more genomes, their corresponding proteins are likely to be functionally linked [17]. This method is helpful for such cases where operon exists, or operon-like clusters are observed. This method is highly accurate but unfortunately has only low coverage due to the very specific orthologs required.

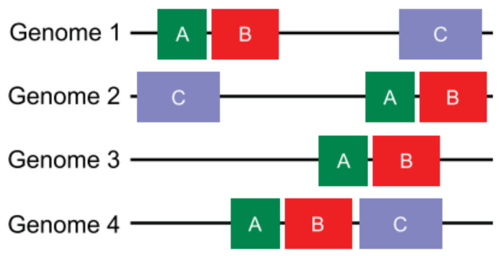


Figure 2.4: The figure shows four hypothetical genomes, containing one or more of the genes A, B, and C. Since the genes A and B are co-localized in multiple genomes (1–4), they are likely to be functionally linked with one another. Source: Raman [79]

2.3 Other methods for prediction of interactions

2.3.1 Docking procedures

Docking procedures use surface complementarity of the proteins and electrostatic forces to fit the proteins' structural models via their interacting surfaces. Such prediction has its limitations due to the limited understanding of the forces involved [23].

2.3.2 Bayesian approach

In Jansen et al. [36] a bayesian network is implemented to predict protein-protein interactions. The method combines and weights genomic features that are not strongly associated with protein-protein interactions. Moreover, it can integrate noisy experimental interaction prediction data to improve the prediction.

2.3.3 Literature mining

Due to the huge amount of scientific papers, it is impossible to keep track of all relevant publications. Usually, protein-protein interactions are recorded in a database. If it is not the case, for example, for very recent studies, literature mining tools can search for records in a large number of scientific papers [13].

2.3.4 Other machine learning and artificial intelligence methods

In recent years protein-protein interaction prediction experienced a boom because of huge advances in machine learning and artificial intelligence in general. The models are trained on enormously big datasets containing various features from the sequential and physicochemical function of proteins and their respective DNA to data from homologous sequences. [18, 19]

2.4 Summary

All of the proposed methods for protein function prediction require the availability of annotated proteins that are in some way (sequentially, structurally) similar to the query protein. Moreover, the methods are not utterly reliable, and a wrong annotation in a database may be further propagated [31].
However, for protein-protein interactions, the conditions are slightly more relaxed. Some similarity-based methods do not require high global similarity but only high local similarity for domain-coding regions. Moreover, modern methods presented in the previous section are using a wide range of features and thus are not entirely dependent on finding highly similar proteins.

3. Prediction of function from PPINs

Having more information about protein-protein interactions than about protein function makes it beneficial to further use the information about protein-protein interactions. This is accomplished in methods based on protein-protein interaction networks. Moreover, PPINs based methods for evaluating the reliability of interaction exist [58, 49], which further enhance the quality of function prediction from PPINs.
Protein function prediction from PPIN also has its limitations, but it complements the previously mentioned methods and can be combined with them. Moreover, it can shed some light on areas where standard methods are insufficient.

3.1 Graph-theoretical background

A graph is a structure for capturing relationships between entities such as people, cities, or proteins in our case. It consists of so-called vertices (also called nodes), representing the entities and edges representing the relationships between the vertices.

Definition 1 (Graph)**.** *A **graph** is a pair $G = (V, E)$, where V is a set of vertices and E a set of edges. $E \subseteq \{(u, v) : u, v \in V\}$.*

We can also assign weights to vertices or edges, meaning that a value is assigned to each vertex or edge. The vertices usually represent some entities, and the edges some relationships between them.

Definition 1 ((Weighted graph)**.** *A **weighted graph** is a graph in which every edge e has a weight w assigned.*

A graph can be directed or undirected. For an undirected graph, en edge can be defined as a set of two vertices $E \subseteq \{\{u, v\} : u, v \in V\}$, whereas for an directed graph, an edge is an ordered pair of vertices $E \subseteq \{(u, v) : u, v \in V\}$. Thus an edge by a directed graph is bi-directional, and by a directed graph, a direction of every edge is defined. An undirected edge can also be interpreted as two directed edges with opposite directions.
Undirected graphs are used when the displayed relationship is symmetric, for example, two proteins interacting. Directed graphs illustrate asymmetric relationships, for example, biological pathways.

3.1.1 Graph representation

Vertices are usually represented as dots or circles, edges as lines, by undirected graphs, or arrows by directed graphs.

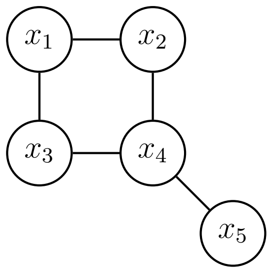


Figure 3.1: Undirected graph G

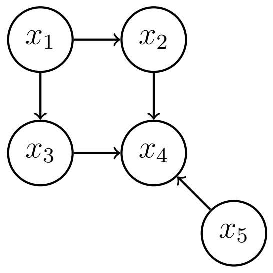


Figure 3.2: Directed graph G

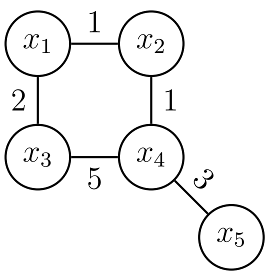


Figure 3.3: Weighted graph G

Note that the way a particular graph is drawn posses no information value.

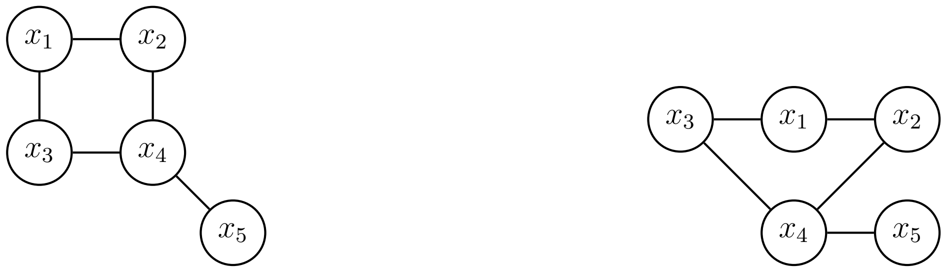


Figure 3.4: Two depictions of the same graph G

We can also encode a graph in an adjacency matrix.

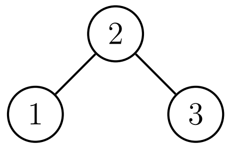
Definition 2 (Adjacency matrix)**.** *An* ***adjacency matrix*** *is a square matrix, which elements indicate whether pairs of vertices are adjacent or not in the graph.*

$$a_{i,j} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \in E(G) \\ 0 & \text{if } \{v_i, v_j\} \notin E(G) \end{cases} \tag{3.1}$$

3.1.2 Subgraph

Usually, we are interested only in some part of the entities and their relations. For this case, we define a subgraph.

Definition 3 (Subgraph)**.** *A* ***subgraph*** *S of a graph G is a graph in which vertices are a subset of the set of vertices in G and edges are a subset of the edge set of G.*



$$A = \begin{array}{c|ccc} & 1 & 2 & 3 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 1 \\ 3 & 0 & 1 & 0 \end{array} \tag{3.2}$$

Figure 3.5: A numbered graph G on the left and its corresponding adjacency matrix on the right

We often want the subgraph to preserve all relationships between its entities. Thus we define an induced subgraph.

Definition 4 (Induced subgraph)**.** *An* ***induced subgraph*** *S of graph G is a graph in which a set of vertices is a subset of the vertices of G, and a set of edges are all those edges in G that connect the vertices of the subgraph.*

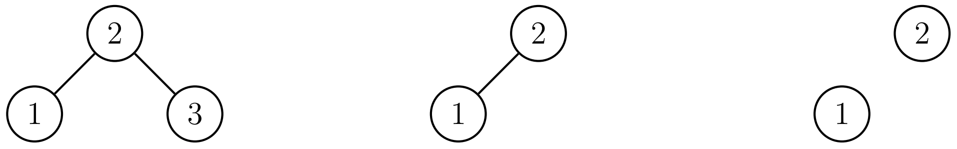


Figure 3.6: Graph G on the left, its induced subgraph S_1 in the middle and its not induced subgraph S_2 on the right.

3.1.3 Graph and subgraph types

We recognize several important types of graphs listed in the following definitions.

Definition 5 (Complete graph)**.** *A graph G is said to be* ***complete****, denoted K_n if there exists an edge between all pairs of vertices.*

A complete subgraph is called clique.

Definition 6 (Clique)**.** *A* ***clique*** *is a complete subgraph of a graph.*

An important subgraph of a graph is a path.

Definition 7 (Path)**.** *A* ***path*** *is a sequence of distinct vertices where an edge connects each consecutive pair, and there are no other edges.*

Usually, we are interested in the shortest path because it represents the distance between two entities.

3.1.4 Graph properites

Definition 8 (Path length)**.** *A* ***length of a path*** *between two vertices v_1, v_2 in a graph G is a sum of the weights of its constituent edges.*

Definition 9 (Shortest path)**.** *A* ***shortest path*** *between two vertices v_1, v_2 in a graph G is a path from v_1 to v_2 with a minimal path length.*

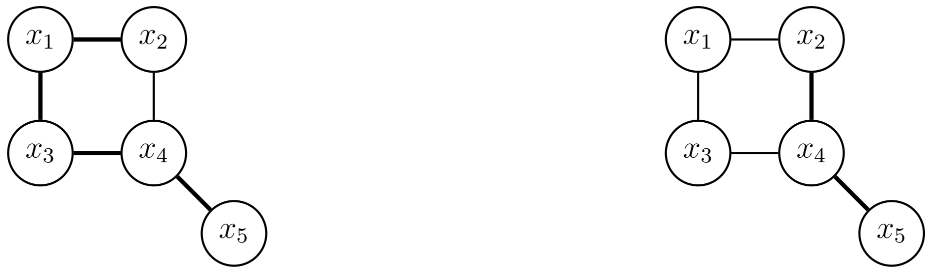


Figure 3.7: A path between x_2, x_5 in the graph G on the left, a shortest path between x_2, x_5 in the graph G on the right. Weight one is assigned to all edges e of the graph G.

We also define the distance between two vertices as the lenght of their shortest path.

Definition 10 (Distance)**.** *A **distance** between two vertices u and v in graph G is the length of the shortest path between them.*

Then we are interested, whether two vertices are in some way connected.

Definition 11 (Connectivity)**.** *Two vertices are **connected** if there is a path between every two vertices.*

This property can be generalized for whole graphs.

Often we are interested in a set of entities that are connected. We call such a set a connected component of a graph.

Definition 12 (Connected component)**.** *A **connected component** of an undirected graph is a connected induced subgraph connected to no other vertices in the graph.*

Note that vertices of each graph can be decomposed into disjoint subsets, each one inducing a connected component.

Definition 13 (Cycle)**.** *A **cycle** is a path in a graph where the starting point is the same as the endpoint.*

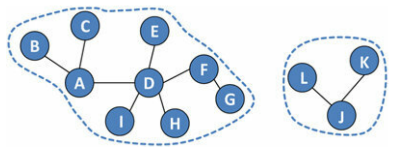


Figure 3.8: A graph with two connected components. Source: Kolb et al. [48]

We can also be interested in how important is a vertex in the sense of how many neighbors it has.

We can examine the vertices that are directly connected to a selected vertex.

Definition 14 (Neighborhood)**.** *A **neighborhood** of a vertex v is the set of all vertices connected to it. A k-**neighbourhood** of vertex v is the set of all vertices which distance from v is at most k.*

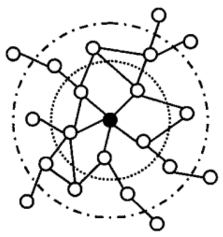


Figure 3.9: Neighborhood levels of the vertex in the center. Source: Hishigaki et al. [32]

Definition 15 (Degree)**.** *A **degree** of a vertex v is the size of its neighborhood. The degree is denoted $deg(v)$.*

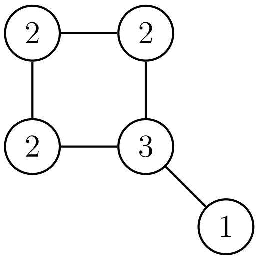


Figure 3.10: Graph with vertices labeled by their degree

The graph can also be examined globally. For example, whether the vertices tend to have only a small number of neighbors or whether they are densely connected.

Definition 16 (Graph density)**.** *A **density**, denoted $den(G)$ of a graph G=(V, E), is defined as the number of its edges $|E|$ divided by its maximum number of edges. For a simple graph that is $|V| \cdot (|V| - 1)/2$*

3.1.5 Networks

Large real-world graphs are called networks. Networks usually represent a large amount of data and relationships between them in the form of a graph. The data may, for example, represent people and relationships between them, cities and roads, or biological pathways [16].

By networks, we usually examine whether some more connected subgraphs are present. For real networks, this may be, for example, roads within a city. To measure how clustered a network is, we use the so-called clustering coefficient [93].

Definition 17 (Clustering)**.** *Networks are said to be highly* ***clustered*** *if two vertices with a common neighbor have a heightened probability also to be neighbors of each other. We can quantify this property by clustering coefficient [93].*

$$C = \frac{3 \cdot (\textit{number of triangles in a graph})}{\textit{number of connected triples}} \tag{3.3}$$

Where triangle is, a cycle of length three and connected triplet is connected induced subgraph consisting of three vertices.

The clustering coefficient is 1 on a complete graph. For the real networks, the clustering coefficient is noticeably higher than for random networks (around $0.1 - 0.5$ [29]).

Dense clusters are called communities.

Definition 18 (Community)**.** *A* ***community*** *is a subset of vertices that are densely connected with each other and sparsely connected to the vertices outside of the community.*

Highly clustered networks are usually community networks.

Definition 19 (Community networks)**.** *A network G is said to have a* ***community structure*** *if it can be divided into communities, so that $V = \bigcup_{i=1}^{n} C_i$, where V is the set of vertices of the network G and C_i is the i-th community of the network G.*

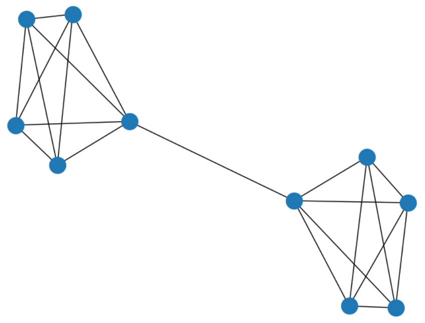


Figure 3.11: A community network

Another common property of networks is the small-world property.

Definition 20 (Small-world property)**.** *Networks with* ***small-world property*** *have a small diameter (maximal distance between two vertices) and are highly clustered.*

3.2 Protein-protein interaction networks

In protein-protein interactions networks, the vertices represent the proteins. An edge between two vertices exists if and only if there is an interaction between the two proteins represented by the two vertices. The networks are usually not oriented because the interaction relation is symmetric, meaning if protein one interacts with protein two, then protein two also interacts with protein one.

3.2.1 Databases

Different institutes have their own databases of protein-protein interactions. However, some of them are deprecated or are using formats that are complicated to parse. Some of them are specialized only for some class of organisms. The most up-to-date database is STRING. In addition to PPI, it also contains a reliability score for each one of them, determined by in-silico methods. The database also has two clustering algorithms implemented and contains additional information about proteins from other databases. Moreover, it is directly embedded in Cytoscape, software for visualization and network analysis.

Database	URL	Description
STRING	`https://string-db.org`	Protein networks based on experimental data and predictions at EMBL. A large number of different organisms. Up to date.
APID	`http://cicblade.dep.usal.es:8080/APID/init.action`	Agile Protein Interaction DataAnalyzer (Cancer Research Center, Salamanca, Spain)
DIP	`https://dip.doe-mbi.ucla.edu/dip/Main.cgi`	Database of Interacting Proteins at UCLA. No species restriction.
MIPS	`https://mips.helmholtz-muenchen.de/proj/ppi/`	Collection of manually curated high-quality mammalian PPI data collected from the scientific literature.
HPDR	`http://www.hprd.org`	The Human Protein Reference Database. Institute of Bioinformatics, India and Johns Hopkins University, USA.

Table 3.1: Databases of PPI. Source:MIPS website

3.3 Principles of function prediction

For this method to be applicable, some of the proteins in the network must be already annotated.
The methods can be generally divided into two categories.

3.3.1 Neighbourhood based methods

Neighborhood-based methods use the fact that the proteins that have a small distance to each other in the protein-protein interaction network often have similar functions [86].

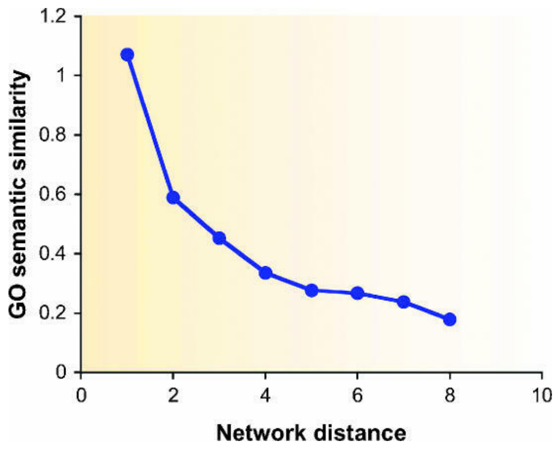


Figure 3.12: Neighbourhood function relationship. Distance in the unweighted network on x-axis, by the shortest path metrics. Average function simmilarity of proteins of specific distance on y-axis. Simmilarity mesured using semantic similarity as introduced in Lord et al. [62]. Source: Sharan et al. [86].

The neighborhood-based methods concentrate on a small neighborhood of each vertex. The most basic approach is introduced in Schwikowski et al. [85], where only neighbors of a vertex are taken into account and the most frequently occurring protein functions are selected. In Chua et al. [14] the frequencies are normalized, and more distant vertices are taken into account.

These methods were the first introduced ones and served as a basis for modern methods; however, they are mostly too inaccurate to be used nowadays. Because of that, we will omit them in the following chapters.

3.3.2 Methods based on detection of communities

Most of the biological processes are carried out by protein complexes [30]. Therefore, in order to identify protein with shared function, the methods aim to identify the protein complexes. As stated in Tong et al. [88], the densely connected regions in PPINs often correspond to protein complexes. In the language of mathematics, the densely connected regions in a network are called communities. The problem of community detection in complex networks is widely studied [46]. Hence the applicability of known algorithms for PPINs and their modifications to reflect the specific nature of the communities of proteins is studied.

To annotate the PPINs, the network is first divided into communities by a clustering algorithm. Then, for each community, the frequencies of functions present in the community are calculated and normalized according to the overall frequency of the particular protein function. The proteins with unknown functions are then annotated with the functions with a frequency above a given threshold.

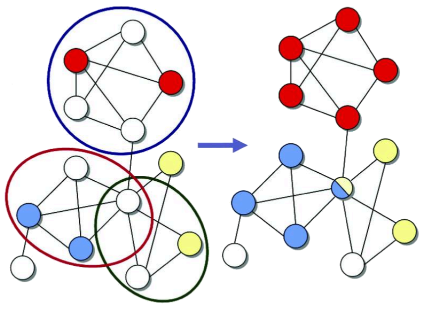


Figure 3.13: Process of communities identification and function assignment. Source: Sharan et al. [86]

3.4 Problems and challenges

3.4.1 Overlapping of communities

It was proved that many proteins are present in more than one complex and hence might have more functions. For example, out of 1,628 proteins in the CYC2008 yeast data set, 207 is present in more than one complex [76]. For many clustering algorithms, this proposes a problem because they cannot assign a vertex to more than one community.

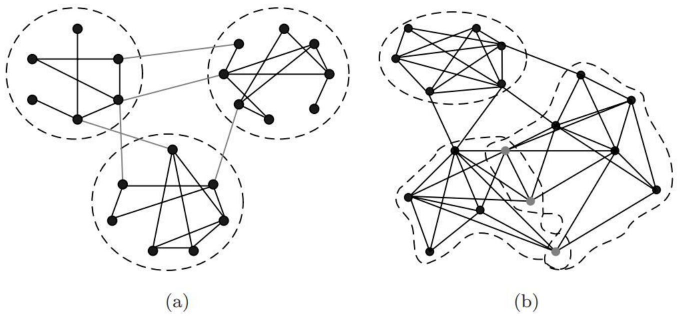


Figure 3.14: Non-overlapping (left) and overlapping communitites (right). Source: Gao et al. [27]

3.4.2 Unreliability of the interaction data

PPI data are often determined using in-silico methods, which are not entirely reliable, as stated in the previous chapter. As a result, many false positives interactions are present in the network, and many actual interactions are missing [72].

Fortunately, there exist topological methods for reliability assessment based on the neighbourhood information [58], distance [49] or clustering [44]. Which can be incorporated into the function prediction algorithms.

3.4.3 Time complexity

PPI networks usually consist of thousands of vertices. Moreover, many clustering algorithms have adjustable variables; this requires multiple runs of the algorithm. Therefore there is a need for a reasonable running time of the algorithms.

3.4.4 Detection of sparse complexes

Many algorithms focus on detecting dense subgraphs of the PPI network. However, the protein complexes are not always dense [59].

3.4.5 Distance metrics

The primary metric of the distance of two proteins in the network is the shortest path metrics; however, due to the small-world property of the protein-protein interactions networks, which causes that every two vertices are relatively near each other, this metric may not be discriminating enough.

For this reason, other metrics have been developed, such as Czekanovski-Dice distance [10] or statistical measure via p-value [84].

4. Algorithms for detection of communities in PPINs

In recent years, many different algorithms for community detection in protein-protein interaction networks have been developed. These can be divided into the following categories. The most used ones from every category are listed.

Approach	Algorithms
Density based	MCODE [6], CFinder [1], ClusterONE [67]
Flow based	MCL[90], Tribe-MCL[22], STM[34]
Core based	COACH[96], CORE[54]
Evolutionary alghorithms	ACO-MAE [37], NACO-FMD[38], NHB-FMD[60]
Hierarchy based	Jerarca[2], HC-PIN[92]
Spectral clustering	ADMSC[35], [77]
Partition based	RNCS[47], Edge-betweeness [21]
Simulated annealing	[91]

Table 4.1: Clustering algorithms

In the following sections, we will present different approaches for the detection of communities in PPINs and evaluate their strengths and weaknesses. In every section, we will generally speak about algorithms using specific approaches and then take a closer look at widely used ones for illustration.

4.1 Density based algorithms

Density-based algorithms search for densely connected subgraphs in the network, often by finding maximal cliques (cliques to which no other vertices can be added without disrupting their cliquishness) in the network and merging those with a lot of common vertices. Therefore they usually explore local properties of the network. Most of them are also able to detect overlapping communities. Unfortunately, these algorithms often fail to recognize sparser communities or connect almost isolated vertices [39].

4.1.1 MCODE

The MCODE algorithm was proposed in Bader and Hogue [6] and was one of the first successful algorithms for the given problem.

The algoritm has three phases.

Phase one - weighting

Definition 21 (k-core). *A **k-core** of a graph G is a subgraph S of minimal degree of k.*

Definition 22 (Highest k-core)**.** *A **highest k-core** of a graph G is a k-core with the highest k in G.*

Definition 23 (Core-clustering coeffitient)**.** *A **core-clustering coefficient** of a vertex v in a graph G is the density of the highest k-core of the immediate neighborhood of v (vertices connected directly to v) including v [6].*

Each vertex v of the graph G is weighted based on the product of its core-clustering coefficient and the k-core of the vertex u, which is an immediate neighbor of v and has the highest k-core among all neighbors of v.
Therefore, the weighting function is based on computing the local density with further boost for densely connected vertices. It is also possible to use a custom weighting function.

Phase two - Molecular complex prediction

A vertex v with the highest weight is selected. Then a recursive algorithm is performed, where all of the neighbors of v, which weight is above a threshold, are visited, and the same happens for their neighbors and so on. The recursive algorithm terminates when no unseen neighbor-vertices above the given threshold exist. The same is performed for the next unseen vertex with the highest score. The whole algorithm terminates when there are no unseen vertices.

Threshold A threshold is a given percentage away from the weight of the seed vertex, that is, the vertex with the highest weight from the first level of the recursive algorithm. Thus the threshold parameter defines the density of the resulting complexes. The closer to the weight of the seed vertex, the smaller and denser the resulting complexes are.

Phase three - postprocessing

Complexes that do not contain at least 2-core are removed, and their vertices marked again as unseen. The algorithm has two modes. In the ”fluff” mode, for each vertex of each community, its unseen neighbors are added to the community belonging to the vertex if the neighborhood density is higher than the given fluff parameter. When a vertex is added to a community, it is not marked as seen and can thus be added in another community. Therefore the resulting communities may overlap.
In the ”haircut” mode, the unseen vertices are removed. It is possible to run both the modes, first, the fluff mode followed by the haircut mode.

Problems

MCODE usually clusters the data in a small number of communities, which results in some large communities [39]. Moreover, the algorithm aims to find densely connected subgraphs, but protein complexes are not always densely interconnected.

4.2 Flow algorithms

Flow algorithms simulate a flow through the network. Through simulation of flows within PPI networks, these algorithms can predict complex network behaviors under stimuli of each vertex in the network [39]. Moreover, the reliability scores of individual interactions can be easily integrated.

To illustrate what a flow is, we can imagine our network as a network of pipes. For our example, we pour an equal amount of liquid in each vertex and assign to each vertex-edge pair what fraction of the total amount of liquid will flow into that particular edge from the vertex. We let the liquid flow between the vertices for some time and observe the amount of liquid flowing through each edge. For the sake of the illustration, we allow the liquid to flow in both directions simultaneously.

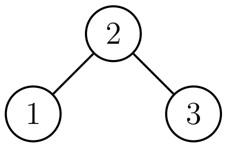
4.2.1 Markov Clustering Algorithm

Mathematical basics

Imagine the network as a network of cities and roads between them. Now imagine traveling between the cities and in each city choosing the next destination from the cities connected to the current city with a road at random. The probabilities of choosing city j when in city i can be encoded in the Markov matrix.

Definition 24 (Markov matrix)**.** *The **Markov matrix** has on the index i, j the probability of selecting the edge $\{v_i, v_j\}$ from all edges directing from v_i, if there exists an edge between v_i and v_j, else 0. The probability of selecting a certain edge incident to one vertex is uniform for each edge incident to the vertex.*

$$m_{i,j} = \begin{cases} \frac{1}{d(v_i)} & \textit{if } \{v_i, v_j\} \in E(G) \\ 0 & \textit{if } \{v_i, v_j\} \notin E(G) \end{cases} \tag{4.1}$$



$$M = \begin{array}{c|ccc} & 1 & 2 & 3 \\ 1 & 0 & 1 & 0 \\ 2 & \frac{1}{2} & 0 & \frac{1}{2} \\ 3 & 0 & 1 & 0 \end{array} \tag{4.2}$$

Figure 4.1: A numbered graph G and its corresponding Markov matrix M

Starting from a city i, after k iterations, one will end up in city j. Such travel is in graph theory called a random walk of length k from vertex i to vertex j. The probability of ending up in city j when starting in city i in the k-th iteration is given by the element $M^k_{j,i}$ of the k-th power of the Markov matrix M.

Algorithm

MCL, proposed in Van Dongen [90], uses the graph clustering paradigm, that *"A random walk in G that visits a dense community will likely not leave the community until many of its vertices have been visited."*[90]. The basic idea behind the algorithm is to simulate a flow within a graph. According to the paradigm, the current will be stronger within the communities and weaker in between them. The algorithm further promotes the flow where the current is strong and weakens the current where the current is weak.

Expansion In the expansion step, the Markov matrix is expanded by taking the e^{th} power of the matrix M. $M := M \cdot M^e$. The element $e_{i,j}$ of the newly expanded matrix M will then be the probability of a random walk from j to i with the length $e \cdot n$ where n is the number of interactions.

Inflation In the inflation step, the differences between currents are further enhanced. For every column of the matrix, each element is taken to the power r, and the column is then normalized by the sum of its elements.
This way, the differences between elements with high value and elements with a low value of the Markov matrix are further promoted.

The algorithm is mathematically very simple and straightforward. First, a Markov matrix is constructed. Then following steps are repeated till convergence.

1. Expansion
2. Inflation

Both these steps can be achieved by simple matrix multiplication. The resulting matrix will have some entries very close to 0. The last step is to remove corresponding edges from the network (i. e. if the value of the element $M_{i,j}$ is close to 0, ve remove the edge between vertex i and vertex j) and identify connected components of the resulting network. Every connected component corresponds to a community.

Advantages

This algorithm is considered very robust, meaning adding or removing a couple of vertices will not significantly alter the structure of the community.
In Pereira-Leal et al. [73] the PPIN is first transformed into a line graph.

Definition 25 (Line graph)**.** *The vertices of the* ***line graph*** *L are the edges of the original graph G, and edges between them exist if and only if the two edges of G are incident.*

This way also more distant vertices are taken into account for each vertex. Moreover, reliability scores of PPIs can also be directly encoded in the Markov matrix, and overlapping communities are detected [73].

Problems

If there is a vertex connected to the rest of the network only by one edge, this vertex will not be added to the neighboring community. Girvan and Newman [29].

4.3 Core based algorithms

Communities in PPINs usually contain a core, that is, a dense subgraph. The core proteins have high function similarity and coexpression. Attached proteins then surround this core [28]. The core-based algorithms are designed to detect communities with this nature. Being specialized in detecting communities in PPIN, this approach has an advantage over other algorithms trying to solve more general clustering problems.

4.3.1 COACH

The COACH algorithm, proposed in Wu et al. [96], is based on identifying graph cores.

First, for each vertex v a graph S_v is constructed. This graph contains the vertex v and all its neighbors with an above-average degree. Each subgraph S_v then proceeds into the core graph removal procedure.

Core graph removal

If the density of the input graph S_v is above a given threshold, the whole graph is returned. Else a dense subgraph of S_v, denoted D_{S_v} is identified, in which all vertices have an above-average degree (average taken from the vertices of S_v). Note that D_{S_v} is not necessarily dense.
The vertices of D_{S_v} are then removed from S_v. This way, This way, $S_v \setminus D_{S_v}$ might fall apart into components, $C_{v,1}, \ldots, C_{v,k}$. The core graph removal procedure is recursively called for each connected component $C_{v,i}$, and a list of subgraphs of every connected component is obtained. For each graph $C^R_{v,1}, \ldots, C^R_{v,\ell}$ form the list, an induced subgraph of S_v is created (denoted $L^R_{v,j}$), containing only vertices of $C^R_{v,j}$ and D_{S_v}. A list of densely connected components $L^R_{v,1}, \ldots, L^R_{v,\ell}$ is returned.

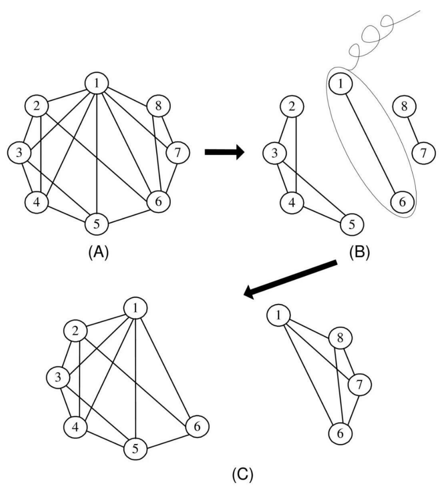


Figure 4.2: (A) shows the core graph of vertex 1's neighborhood graph, denoted as S_1. In (B), the core vertices of S_1, 1, 6, are removed from S_1 and two connected components are left in the remaining graph. In (C), 1, 6 are added back into each connected component. Two subgraphs with vertices 1,2,3,4,5,6 and 1,6,7,8 are obtained and finally returned. Source: Wu et al. [96]

For each graph L_i from the list, first, its vertices with a small degree are removed until the density of L_i is above a threshold. Then vertices from the neighborhood of L_i are added starting with the one with a maximal degree until no such vertices exist and the density is above the threshold.
This way, a lot of densely connected overlapping communities arise. These communities are maximal, meaning adding another vertex would disrupt the density condition.
The communities are added to the global list of all communities, and for each of them, redundant communities are filtered as the final step.

Redundancy filtering

For each newly identified community, the most similar community from the global list of communities is found. The similarity of communities A and B is defined as:

$$NA(A, B) = \frac{|V_A \bigcap V_B|^2}{|V_A| \cdot |V_B|} \tag{4.3}$$

Where V_i are the vertices of the community i.

If their similarity is above a threshold, the community with lower $den(G) \cdot |V_G|$ is deleted from the list of all detected communities. Otherwise, both are kept on the list. This algorithm, therefore, can detect overlapping communities.

4.4 Evolutionary algorithms

4.4.1 Basic concepts

In evolutionary algorithms, a population of individuals is created. The representation of individuals is adapted to the specific problem. Like in nature, the

individuals are mating and ”crossing over their genes,” which results in a new, possibly better generation of individuals. There exists a fitness function computable for each individual that is maximized. Individuals with higher fitness have a higher probability of being selected from the mating pool. The selection, mutation, and crossover are repeated for many generations.
By decoding the individual with the highest fitness, we should obtain a (suboptimal)solution for our problem.

This approach can be illustrated on a simple example, where we are trying to maximize a sum of n numbers from $1 - 9$. In this example, we encode the individual as a sequence of numbers, its fitness as a sum of the numbers, mutation as substituting a number on a random index in the sequence for another number, and crossover as switching the numbers on the same indices between two individuals. (Proposed operators are not optimal for the given problem but are easy to understand). Due to the selection, the individuals containing higher numbers will prevail after a couple of generations.

Traveling salesman problem

A typical problem of evolutionary algorithms is the traveling salesman problem (TSP). It is inspired by a salesman trying to travel through all cities on his map while optimizing the length of his way. For a larger number of cities, this problem is not solvable analytically in a reasonably short time.

4.4.2 Application to PPINs

Further research of PPINs communities will possibly result in a complex set of rules for their possible structure, which would be hard to incorporate in the standard algorithms. For evolutionary algorithms, it may be possible to partially define these constraints in the fitness function. Unfortunately, evolutionary algorithms are often very computationally demanding.

4.4.3 Ant Colony Optimization with Multi-Agent Evolution

Ant Colony Optimization with Multi-Agent Evolution (ACO-MAE) algorithm, introduced in Ji et al. [37] was inspired by colonies of ants, led by pheromones, which can find the shortest path to resources.

First, the edges of the PPIN are weighted

Weighting the edges

We will weight the edges according to the distance of their constituent vertices. The aim is to transform the problem to TSP, by assigning low ”distances” (like the distances between cities) to those proteins, that have a lot of common neighbours.

For two proteins is their ”distance” computed as follows [38]:

$$d_{i,j} = \frac{(Int(i) \bigcup Int(j)) - (Int(i) \bigcap Int(j))}{(Int(i) \bigcup Int(j)) + (Int(i) \bigcap Int(j))} \tag{4.4}$$

where $Int(i)$ is the set of neighbours of i plus the protein i. The proteins that have a similar neighborhood will have a small distance, as they will have a large intersection.

Then a population of individuals is created with the following encoding of each individual.

Encoding of an individual

The proteins are numbered. If a number x is i-th in the sequence, there is a path from protein x to protein i.

After the algorithm is finished, we make a graph G from all vertices without any edges. Then we will add edges according to the numbers in the encoding sequence. Then we identify the connected components of the graph G. Every connected component corresponds to one community.

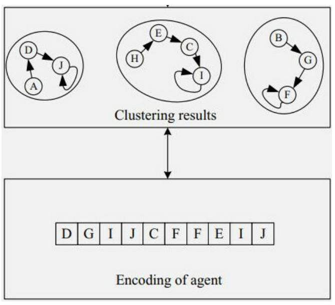


Figure 4.3: Encoding of the individual at the bottom, decoding of the individual at the top. Source: Ji et al. [37]

Fitness of an individual

Each position of the individual, a value is assigned, corresponding to the distance between the index protein and the protein on the particular position. The resulting fitness is the negative of the sum of these values. We typically want to maximize fitness in evolutionary algorithms, therefore, minimize the distance.

The fitness resembles a ”distance” that the individuum travels between the proteins.

Selection

The individuals are placed on a spheric grid to ensure their local perceptivity. Every individual, therefore, has only 4 neighbors. For every individual i, we evaluate its fitness and the fitness of its neighbors. If any of its neighbors have higher fitness, one of two things happens with a certain probability. The neighbor j with the highest fitness will replace i, and the protein on the position with the largest distance in the encoding sequence of j will be replaced with a protein on the respective position from the encoding sequence of i, or the new individual will for every position select value form the individual (i or j) with smaller distance on that position. Note that this way, smaller isolated paths may arise.

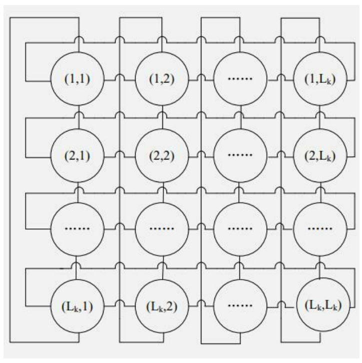


Figure 4.4: Grid-like evolutionary environment. Source: Ji et al. [37]

Mutation and crossover

For mutation, random position i in the sequence is selected, and the protein on position i is replaced with a protein from another neighbor of protein i. For crossover, a protein is selected randomly from one of the two neighboring individuals for each position.

Postprocessing

After running the algorithm for many generations, for the individual, with the highest fitness, if the distance between the two following proteins is above the given threshold, we switch the path between them for a self-loop.
This individuum is then returned and decoded.

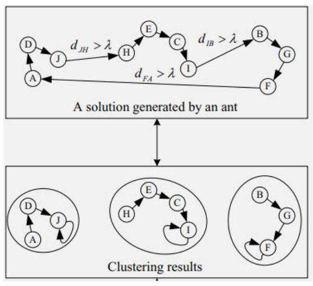


Figure 4.5: The process of creating self-loops from edges representing interactions of proteins with very different neighborhoods. Source: Ji et al. [37]

4.5 Hierarchy based algorithms

The main advantage of hierarchical clustering is that it in some way resembles the process of evolution of proteins and their interactions [39].
The algorithms are based on the premise that members of one community are likely to have similar shortest-path-distance profiles [81]. For many hierarchy-based algorithms, using simple shortest path metrics proposes a problem because distances between many protein pairs are identical [4]. Using the shortest path metric would make the resulting profiles too noisy. For this reason, other metrics are used.

Unfortunately, most of these algorithms are not able to detect overlapping communities.

4.5.1 Jerarca

Jerarca algorithm, proposed in Aldecoa and Marín [2] can be divided into three phases.

Phase one - secondary distance matrix

In the newest version of the algorithm, all maximal cliques (cliques to which no other vertices can be added without disrupting their cliquishness) in the network are detected as primary communities. This is generally an NP-complete problem, but since PPINs are sparse networks, it is still computable in a reasonable time. Then the distance between two vertices is then computed as:

$$d_{i,j} = \frac{|C| - |C_i \bigcap C_j|}{|C|} \tag{4.5}$$

Where $|C|$ is the number of detected communities and $|C_i \bigcap C_j|$ the number of vertices that the communities i and j have in common.
This way, a distance matrix D is constructed, where on index $D_{i,j}$ in the distance between vertices i and j.

Phase two - tree construction

From the distance matrix, D a dendrogram (a diagram illustrating the hierarchical relationship between objects) is constructed with UPGMA [87] or Neighbor-joining algorithm [83]. Communities are established by scanning the dendrogram starting from its root. Each dichotomy generates an alternative community. Since the Neighbor-joining algorithm generates an unrooted tree, the middle point of the tree is used as the root.

Phase three - community structure evaluation

Each possible partition into communities is evaluated according to two criteria.

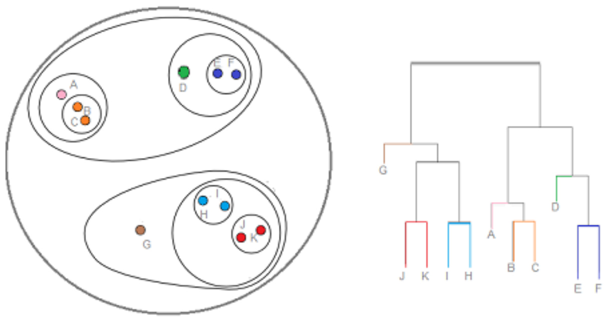


Figure 4.6: Dendrogram with colored leaves correspondong to detected communitites. New community is constructed for each dichotomy. Source: statisticshowto

Modularity First a matrix E is constructed, where on position $E_{i,j}$ is the fraction of all edges in the network that link vertices in community i to vertices in community j. Modularity is then defined as:

$$Q = Tr(E) - ||E^2|| \tag{4.6}$$

Where $||X||$ is the sum of all elements of the matrix X [68].

Definition 26 (Trace)**.** *The* ***trace*** *of a matrix M denoted $Tr(M)$ is the sum of its diagonal.*

This equation measures the distributions of links within and between communities in a particular partition, compared to the expected number of connections for the same degree distribution [2].

H-index

$$H = -\log \sum_{j=p}^{\min(M,n)} \frac{\binom{M}{j}\binom{F-M}{n-j}}{\binom{F}{n}} \tag{4.7}$$

Where p is the total number of direct intracommunity interactions detected, M is the maximum possible number of intracommunity direct interactions, F is the maximum possible number of edges in the network, and n is the number of edges in the network.

H-index aims to select the partition that generates a distribution that maximizes the proportion of intracommunity direct interactions while minimizing the proportion of intercommunity direct interactions [4].

The partition with the maximal product of H-index and Modularity is then selected.

Advantages

The UVCluster algorithm, used in the first phase for distance matrix computation, can easily incorporate biological data and, this way, adjust the distance matrix [4].

4.6 Label propagation algorithms

In recent years, the label propagation algorithms proved to be very suitable for predicting function from PPINs, and thus are now one of the leading algorithms [41, 97].

Lable propagation algorithms, first proposed in Raghavan et al. [78] are based on a straightforward idea. First, a distinct label is assigned to each vertex. Then for each vertex v, the most occurring label in the immediate neighborhood is assigned v. The vertices are iterated in random order. If there are more labels with the maximal frequency, one is chosen randomly. The relabeling of all vertices is repeated until all the vertices have the label that is one of the labels with maximal frequency in their neighborhood. The algorithm is nearly linear and thus runs fast also on huge datasets.

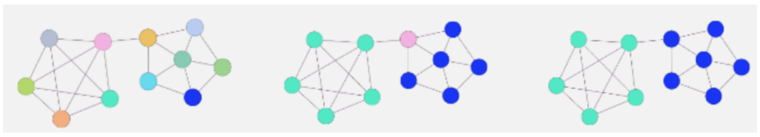


Figure 4.7: Label propagation on a network, (A) the original network, (B) an intermediate result, and (C) final communities. Source: [41]

4.6.1 Link-driven label propagation algorithm

The basic algorithm was modified in many ways to enhance the prediction reliability for PPINs and to be able to operate on noisy data. The most recent paper, relating to the usage of label propagation algorithms in PPINs, introduced a link-driven label propagation algorithm(LLPA) [41] labels the edges, not the vertices, and weights the edges by the reliability of the respective protein interactions. The edges are then not iterated in random order but with descending reliability.

One of the basic relabeling algorithms' main problems is that it chooses randomly from the labels with maximal frequency. In the LLPA, the selection of the label is more complex. For every edge incident to the currently labeled edge, a value is computed as the product of the reliability of the edge and its similarity to the labeled edge. These values are then summed for every label category, and the label with the highest value is assigned.

5. Evaluation

The evaluation of the algorithms for community detection in PPINs proposes a difficulty for the following reasons. Firstly, the performance differs significantly between different datasets, as we will see. Secondly, authors of a new algorithm often make a performance comparison with other algorithms. This way, the parameters of the algorithms are not optimized carefully. Therefore an evaluation from an independent paper [39] is taken. Another problem is finding reliable metrics for evaluation. These metrics should be able to evaluate the quality of the algorithms given the experimentally obtained incomplete information.

5.1 Properties

First, we evaluate properties independent of real-world data.

5.1.1 Detection of overlapping communities

As stated in previous chapters, most proteins do not have only one function. Hence the functional modules are overlapping. Therefore, it is advantageous for the algorithms to detect overlapping communities.

5.1.2 Robustness

Robustness addresses the algorithm's ability to perform well under the addition and deletion of edges. This property is essential for real-world data because the PPI data contain false positives and false negatives. The experimentally tested robustness is taken from Ji et al. [39].

5.1.3 Incorporation of the reliability information

Because of the existence of methods that can evaluate the reliability of predicted interactions, it is favorable to be able to incorporate the reliability information data into our algorithms. The extension of the reliability information was introduced for some of the examined algorithms.

5.1.4 Properties of algorithms

Method	Overlap	Robustness	Reliability incorp.
MCODE	yes	good	No
MCL	no[1]	good	Yes
COACH	yes	good	No
ACO-MAE	no	good	No
Jerarca	no	fair	Yes

[1] But implemented in a later version

Table 5.1: Evaluation of data-independent properties

5.2 Quality estimation on experimental data

Now, we evaluate the performance of the algorithms on real-world data. These results would vary depending on the data set. The performance was tested by Ji et al. [39] on data from two databases, DIP and MIPS, on proteins with known functions and known complexes. The MIPS dataset is much larger and possibly noisier. Unfortunately, the experimentally obtained complexes and the functional annotation are also not utterly reliable, and so some existing complexes were not detected, functions not discovered, and some false positives are present.[55]

Our dataset contains proteins that were experimentally examined, and their complexes were detected. The experimental detected complexes are denoted as **real communities**.

5.2.1 Evaluation metrics

For evaluation, real communities are compared with the predicted. Unlike the real complexes, the communities obtained from some of the algorithms are non-overlapping.

Matching score

A matching score between real community r and computationally predicted community p is defined as:

$$NA(r,p) = \frac{|V_r \bigcap V_p|^2}{|V_r| \cdot |V_p|} \tag{5.1}$$

Communities that have a matching score above 0.25 are considered to be matching.

Recall

Recall meassures the ratio of matching communities to all predicted communities.

$$R = \frac{N_p}{|P|} \tag{5.2}$$

Where N_p is the number of predicted communities matching at least one real community and $|P|$ is the total number of predicted communities.

Precision

Precision meassures the ratio of real commmuntites, that were predicted by the algorithm to all real communities.

$$P = \frac{N_r}{|R|} \tag{5.3}$$

Where N_r is the number of real communities matching at least one predicted community and $|R|$ is the total number of real communities.

F-measure

F-measure is defined as the harmonic mean of recall and precision.

$$F = 2 \cdot \frac{R \cdot P}{R + P} \tag{5.4}$$

Sensitivity

The sensitivity indicates what percentage of proteins of each protein complex is found in a real complex, and thus the correct function was highly probably assigned to those proteins. However, the sensitivity metric has its limitations. If an algorithm predicts a giant community, covering many real communities, this community will obtain a high score [55].

$$S = \frac{\sum_{i=1}^{|R|} \max_j(T_{ij})}{\sum_{i=1}^{|R|} N_i} \tag{5.5}$$

Where $|R|$ is the number of real communities, T_{ij} is the size of the intersection of the set of proteins in i-th real community, and j-th predicted community and N_i is the size of the i-th real community.

Positive predictive value

The positive predictive value indicates if the predicted communities have their vertices scattered among multiple communities or if the vertices are concentrated in one community. This metric's value depends on the experimental data because, by strongly overlapping real communities, the value will naturally be lower.

$$PPV = \frac{\sum_{j=1}^{|P|} \max_i(T_{ij})}{\sum_{j=1}^{|P|} T_{*j}} \tag{5.6}$$

Where $|P|$ is the number of predicted communities and T_{*j} is the sum of the size of the intersections of predicted community j with all real communities.

Accuracy

Accuracy is defined as the geometric mean of sensitivity and positive predictive value.

$$A = \sqrt{S \cdot PPV} \tag{5.7}$$

p-Value

The p-Value metric, proposed in Bu et al. [11] indicates the statistical significance of the occurrence of a predicted protein community concerning given functional annotation. This metric also works directly with the function, not only with communities. The functional homogeneity of a predicted community is the smallest p-value over all the possible functional groups. A predicted complex with a low functional homogeneity is likely to be an actual protein complex [55]. Hence the smaller p-Value is, the better.

$$\text{p - Value} = 1 - \sum_{i=0}^{k-1} \frac{\binom{|F|}{i}\binom{|V|-|F|}{|C|-i}}{\binom{|V|}{|C|}} \tag{5.8}$$

Where predicted complex C contains k proteins in the functional group F and $|V|$ is the number of proteins in the whole PPIN.

The use of p-Value is generally limited as it requires rich molecular function information that is often not present in the data sets.

5.2.2 Limitations of evaluation metrics

None of the above-described metrics is by no means absolutely accurate. We already proposed the limitations of sensitivity and positive predictive value. A test was made on proteins from overlapping complexes from the MIPS database. When the real complexes were evaluated against themselves, the resulting PPV was only 0.772 instead of 1 [55]. In addition, the metrics assume that all protein communities are known. The reality is far from that. Thus if an unknown but real community is predicted, the metrics will regard it as false positive [55].

5.2.3 Performance on experimental data

Description of the datasets

Two publicly available yeast PPI data sets were employed. The DIP dataset[1] contains 2.526 proteins and 5.949 interactions. The MIPS dataset[2] contains 4.545 proteins and 12.318 interactions. The set of known protein complexes was taken from three different database sources and a set of known protein complexes, presented in Friedel et al. [26].

Performance on the DIP dataset

[1]No more available, available here in the past

[2]No more available, available here in the past

Method	Precision	Recall	F-measure	Time (s)
MCODE	**0.6364**	0.2266	0.3342	3
MCL	0.356	0.3879	0.3717	**1**
COACH	0.5038	0.5	**0.5019**	2
ACO-MAE	0.4224	0.486	0.452	693
Jerarca	0.4286	**0.5093**	0.4655	268

Table 5.2: First part of the evaluation of the introduced algorithms on the DIP dataset of protein-protein interactions.

Method	Sensitivity	PPV	Accuracy	p - Value
MCODE	0.1646	0.2852	0.2166	$4.99 \cdot 10^{-24}$
MCL	0.302	0.3031	0.3026	$\mathbf{1.89 \cdot 10^{-44}}$
COACH	0.2568	0.3065	0.2805	$3.05 \cdot 10^{-37}$
ACO-MAE	**0.3076**	**0.3249**	**0.3161**	$2.05 \cdot 10^{-36}$
Jerarca	0.3034	0.3187	0.3110	$2.65 \cdot 10^{-33}$

Table 5.3: Second part of the evaluation of the introduced algorithms on the DIP dataset of protein-protein interactions.

Performance on the MIPS dataset

Method	Precision	Recall	F-measure	Time (s)
MCODE	**0.3614**	0.1285	0.1896	7
MCL	0.1551	0.3224	0.2095	**2**
COACH	0.3006	0.3201	**0.3132**	8
ACO-MAE	0.2208	**0.3598**	0.2844	2733
Jerarca	0.1632	0.3224	0.2167	1319

Table 5.4: First part of the evaluation of the introduced algorithms on the MIPS dataset of protein-protein interactions.

Method	Sensitivity	PPV	Accuracy	p - Value
MCODE	0.1247	0.2622	0.1808	$1.19 \cdot 10^{-27}$
MCL	0.2201	0.2913	0.2532	$\mathbf{1.04 \cdot 10^{-33}}$
COACH	0.2078	0.2365	0.2217	$7.79 \cdot 10^{-33}$
ACO-MAE	0.2091	**0.3411**	**0.2671**	$1.47 \cdot 10^{-32}$
Jerarca	**0.2328**	0.2986	0.2636	$7.53 \cdot 10^{-32}$

Table 5.5: Second part of the evaluation of the introduced algorithms on the MIPS dataset of protein-protein interactions.

Data visualisation

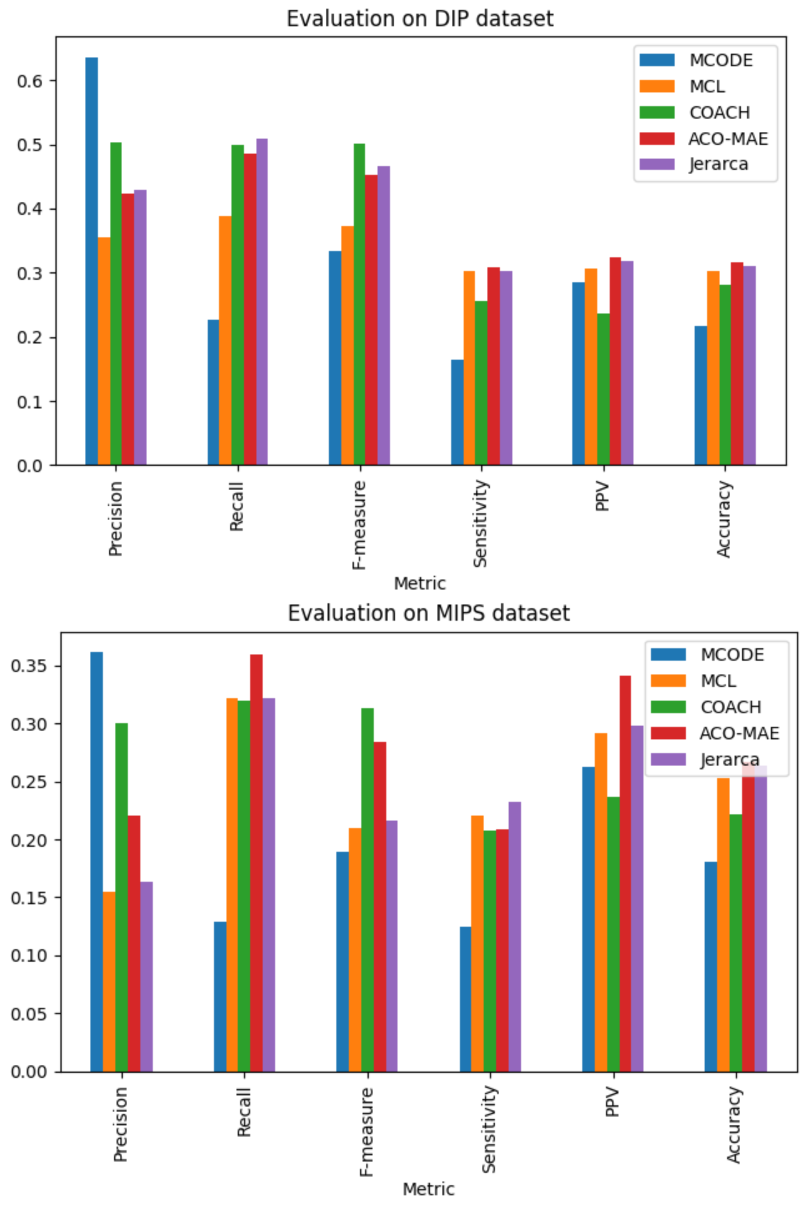


Figure 5.1: Comparision of the introduced algorithms on the DIP and MIPS datasets. Data source: Ji et al. [39]

5.3 Verbal evaluation

5.3.1 Dataset difference

From the performance difference between the DIP dataset and the MIPS dataset, it can be inferred that the prediction quality is strongly related to the data provided. However, for most of the algorithms and metrics, the relative performance of the algorithms has not changed significantly between the two datasets.

5.3.2 Comparision of algorithms

Most surprising is the high precision of the MCODE algorithm, despite its overall low performance. As stated in previous section, the MCODE algorithm tends to predict large communities. These large communities may then contain multiple real complexes while being small enough to pass the lower bound for the matching score. However, this would possibly have a negative effect on the positive predictive value and a positive effect on the sensitivity, which is not the case.
For both datasets, obtained the MCODE algorithm also the smallest p-value, which indicates poor ability to assign a function to proteins correctly.

The F-measure of the COACH algorithm is overall good in comparison with other algorithms, but the accuracy is significantly lower, at least for the MIPS dataset. The reason for this may be that the threshold 0.25 is low enough for some communities to match, but the accuracy takes the ratio of correctly predicted vertices into account, and the lower performance is detected.

The ACO-MAE algorithm has a great overall performance; unfortunately, it is very slow.
The same applies to Jerarca, which also performs very well except for the precision for the MIPS dataset.

The MCL algorithm suffers from low precision; however, its accuracy is comparable with the best-proposed algorithms, it has the best p-value for both datasets, and it is very fast and easy to implement.

The LLPA algorithm was published after Ji et al. [39]. Therefore it was not tested along with other proposed algorithms. However, in the paper introducing the LLPA algorithm, an comparison with other algorithms is published. One of them is the CFinder [1], a density-based algorithm based on maximal cliques, which performance is comparable with algorithms such as Jerarca [39]. While CFinder reached an F-measure score around 0.58, for LLPA, it was around 0.71[39]. However, the dataset was noticeably smaller and probably less noisy.

5.3.3 Summary

Every introduced algorithm has its limitations. They all perform significantly worse on the bigger dataset with possibly noisier data. However, their real performance cannot be calculated because of the false positives and negatives in the experimentally measured protein complexes. The algorithms probably predicted some complexes that were not detected by the experimental methods. Moreover, the metrics for the evaluation also have their limitations, which was visible on the very different scores for the MCODE algorithm.

Conclusion

The genome annotation task is, even to this day, far from being solved. Standard methods for protein function prediction were presented, and their limitations were outlined. A graph-theoretical approach acquiring functional information from protein-protein interaction networks was introduced.
In contrast to standard methods, this approach does not concentrate on the sequence or structure of the proteins but the interactions between them; moreover, the algorithms operate on the whole PPINs, taking global information of the whole interactome into account. Hence, the graph-theoretical methods based on the PPINs complement the standard approaches and can shed some light on areas where standard methods are insufficient.

We identified the noisiness of the protein interactions data, overlapping of the protein complexes, and the time complexity as the main challenges of these methods. We described different types of community-detection-based algorithms, outlined their main ideas, and provided an example algorithm from each category. We then compared these algorithms, assessed their ability to overcome proposed challenges, and evaluated their overall performance. We found that not only designing reliable algorithms but also their evaluation presents a challenge.

From the evaluation results, it can be inferred that the graph-theoretical approach cannot solve the problem of function annotation of proteins; however, it can undoubtedly serve as a complement to other methods of function prediction. Moreover, new algorithms are being developed every year with increasing performance.

Bibliography

[1] Balázs Adamcsek, Gergely Palla, Illés J Farkas, Imre Derényi, and Tamás Vicsek. Cfinder: locating cliques and overlapping modules in biological networks. *Bioinformatics*, 22(8):1021–1023, 2006.

[2] Rodrigo Aldecoa and Ignacio Marín. Jerarca: Efficient analysis of complex networks using hierarchical clustering. *PloS one*, 5(7):e11585, 2010.

[3] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[4] Vicente Arnau, Sergio Mars, and Ignacio Marín. Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21(3):364–378, 2005.

[5] A Selim Aytuna, Attila Gursoy, and Ozlem Keskin. Prediction of protein–protein interactions by combining structure and sequence conservation in protein interfaces. *Bioinformatics*, 21(12):2850–2855, 2005.

[6] Gary D Bader and Christopher WV Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC bioinformatics*, 4(1):1–27, 2003.

[7] Daniel Barker and Mark Pagel. Predicting functional gene links from phylogenetic-statistical analyses of whole genomes. *PLoS computational biology*, 1(1):e3, 2005.

[8] Paul L Bartel, Jennifer A Roecklein, Dhruba SenGupta, and Stanley Fields. A protein linkage map of escherichia coli bacteriophage t7. *Nature genetics*, 12(1):72–77, 1996.

[9] Steven E Brenner, Cyrus Chothia, and Tim JP Hubbard. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proceedings of the National Academy of Sciences*, 95(11):6073–6078, 1998.

[10] Christine Brun, François Chevenet, David Martin, Jérôme Wojcik, Alain Guénoche, and Bernard Jacq. Functional classification of proteins for the prediction of cellular function from a protein-protein interaction network. *Genome biology*, 5(1):1–13, 2003.

[11] Dongbo Bu, Yi Zhao, Lun Cai, Hong Xue, Xiaopeng Zhu, Hongchao Lu, Jingfen Zhang, Shiwei Sun, Lunjiang Ling, Nan Zhang, et al. Topological structure analysis of the protein–protein interaction network in budding yeast. *Nucleic acids research*, 31(9):2443–2450, 2003.

[12] Pea Carninci, T Kasukawa, S Katayama, J Gough, MC Frith, Norihiro Maeda, Rieko Oyama, T Ravasi, B Lenhard, C Wells, et al. The transcriptional landscape of the mammalian genome. *science*, 309(5740):1559–1563, 2005.

[13] Sung-Pil Choi. Extraction of protein–protein interactions (ppis) from the literature by deep convolutional neural networks with various feature embeddings. *Journal of Information Science*, 44(1):60–73, 2018.

[14] Hon Nian Chua, Wing-Kin Sung, and Limsoon Wong. Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions. *Bioinformatics*, 22(13):1623–1630, 2006.

[15] Michele Clamp, Ben Fry, Mike Kamal, Xiaohui Xie, James Cuff, Michael F Lin, Manolis Kellis, Kerstin Lindblad-Toh, and Eric S Lander. Distinguishing protein-coding and noncoding genes in the human genome. *Proceedings of the National Academy of Sciences*, 104(49):19428–19433, 2007.

[16] Luciano da F Costa, Alexandre Evuskoff, Giuseppe Mangioni, and Ronaldo Menezes. *Complex Networks: Second International Workshop, CompleNet 2010, Rio de Janeiro, Brazil, October 13-15, 2010, Revised Selected Papers*, volume 116. Springer Science & Business Media, 2011.

[17] Thomas Dandekar, Berend Snel, Martijn Huynen, and Peer Bork. Conservation of gene order: a fingerprint of proteins that physically interact. *Trends in biochemical sciences*, 23(9):324–328, 1998.

[18] Subhrangshu Das and Saikat Chakrabarti. Classification and prediction of protein–protein interaction interface using machine learning algorithm. *Scientific reports*, 11(1):1–12, 2021.

[19] Xiuquan Du, Shiwei Sun, Changlin Hu, Yu Yao, Yuanting Yan, and Yanping Zhang. Deepppi: boosting prediction of protein–protein interactions with deep neural networks. *Journal of chemical information and modeling*, 57 (6):1499–1510, 2017.

[20] Joe Dundas, Zheng Ouyang, Jeffery Tseng, Andrew Binkowski, Yaron Turpaz, and Jie Liang. Castp: computed atlas of surface topography of proteins with structural and topographical mapping of functionally annotated residues. *Nucleic acids research*, 34(suppl_2):W116–W118, 2006.

[21] Ruth Dunn, Frank Dudbridge, and Christopher M Sanderson. The use of edge-betweenness clustering to investigate biological function in protein interaction networks. *BMC bioinformatics*, 6(1):1–14, 2005.

[22] Anton J Enright, Stijn Van Dongen, and Christos A Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic acids research*, 30(7):1575–1584, 2002.

[23] Iakes Ezkurdia, Lisa Bartoli, Piero Fariselli, Rita Casadio, Alfonso Valencia, and Michael L Tress. Progress and challenges in predicting protein–protein interaction sites. *Briefings in bioinformatics*, 10(3):233–246, 2009.

[24] András Fiser and Andrej Šali. Modeller: generation and refinement of homology-based protein structure models. *Methods in enzymology*, 374: 461–491, 2003.

[25] Iddo Friedberg. Automated protein function prediction—the genomic challenge. *Briefings in bioinformatics*, 7(3):225–242, 2006.

[26] Caroline C Friedel, Jan Krumsiek, and Ralf Zimmer. Bootstrapping the interactome: unsupervised identification of protein complexes in yeast. In *Annual International Conference on Research in Computational Molecular Biology*, pages 3–16. Springer, 2008.

[27] Lin Gao, Peng-Gang Sun, and Jia Song. Clustering algorithms for detecting functional modules in protein interaction networks. *Journal of Bioinformatics and Computational Biology*, 7(01):217–242, 2009.

[28] Anne-Claude Gavin, Patrick Aloy, Paola Grandi, Roland Krause, Markus Boesche, Martina Marzioch, Christina Rau, Lars Juhl Jensen, Sonja Bastuck, Birgit Dümpelfeld, et al. Proteome survey reveals modularity of the yeast cell machinery. *Nature*, 440(7084):631–636, 2006.

[29] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[30] Michal Gorka, Corné Swart, Beata Siemiatkowska, Silvia Martínez-Jaime, Aleksandra Skirycz, Sebastian Streb, and Alexander Graf. Protein complex identification and quantitative complexome by cn-page. *Scientific reports*, 9(1):1–14, 2019.

[31] Troy Hawkins and Daisuke Kihara. Function prediction of uncharacterized proteins. *Journal of bioinformatics and computational biology*, 5(01):1–30, 2007.

[32] Haretsugu Hishigaki, Kenta Nakai, Toshihide Ono, Akira Tanigami, and Toshihisa Takagi. Assessment of prediction accuracy of protein function from protein–protein interaction data. *Yeast*, 18(6):523–531, 2001.

[33] Jaime Huerta-Cepas, Kristoffer Forslund, Luis Pedro Coelho, Damian Szklarczyk, Lars Juhl Jensen, Christian Von Mering, and Peer Bork. Fast genome-wide functional annotation through orthology assignment by eggnog-mapper. *Molecular biology and evolution*, 34(8):2115–2122, 2017.

[34] Woochang Hwang, Young-Rae Cho, Aidong Zhang, and Murali Ramanathan. A novel functional module detection algorithm for protein-protein interaction networks. *Algorithms for Molecular Biology*, 1(1):1–11, 2006.

[35] Kentaro Inoue, Weijiang Li, and Hiroyuki Kurata. Diffusion model based spectral clustering for protein-protein interaction networks. *PloS one*, 5(9): e12623, 2010.

[36] Ronald Jansen, Haiyuan Yu, Dov Greenbaum, Yuval Kluger, Nevan J Krogan, Sambath Chung, Andrew Emili, Michael Snyder, Jack F Greenblatt, and Mark Gerstein. A bayesian networks approach for predicting protein-protein interactions from genomic data. *science*, 302(5644):449–453, 2003.

[37] Junzhong Ji, Zhijun Liu, Aidong Zhang, Lang Jiao, and Chunnian Liu. Ant colony optimization with multi-agent evolution for detecting functional modules in protein-protein interaction networks. In *International Conference on Information Computing and Applications*, pages 445–453. Springer, 2012.

[38] Junzhong Ji, Zhijun Liu, Aidong Zhang, Lang Jiao, and Chunnian Liu. Improved ant colony optimization for detecting functional modules in protein-protein interaction networks. In *International Conference on Information Computing and Applications*, pages 404–413. Springer, 2012.

[39] Junzhong Ji, Aidong Zhang, Chunnian Liu, Xiaomei Quan, and Zhijun Liu. Survey: Functional module detection from protein-protein interaction networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(2): 261–277, 2012.

[40] Peilin Jia, Siyuan Zheng, Jirong Long, Wei Zheng, and Zhongming Zhao. dmgwas: dense module searching for genome-wide association studies in protein–protein interaction networks. *Bioinformatics*, 27(1):95–102, 2011.

[41] Hao Jiang, Fei Zhan, Congtao Wang, Jianfeng Qiu, Yansen Su, Chunhou Zheng, Xingyi Zhang, and Xiangxiang Zeng. A robust algorithm based on link label propagation for identifying functional modules from protein-protein interaction networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2020.

[42] Richard B Jones, Andrew Gordus, Jordan A Krall, and Gavin MacBeath. A quantitative protein interaction network for the erbb receptors using protein microarrays. *Nature*, 439(7073):168–174, 2006.

[43] Trupti Joshi and Dong Xu. Quantitative assessment of relationship between sequence similarity and function similarity. *BMC genomics*, 8(1):1–10, 2007.

[44] Atanas Kamburov, Arndt Grossmann, Ralf Herwig, and Ulrich Stelzl. Cluster-based assessment of protein-protein interaction confidence. *BMC bioinformatics*, 13(1):1–13, 2012.

[45] Kamil Khafizov, Carlos Madrid-Aliste, Steven C Almo, and Andras Fiser. Trends in structural coverage of the protein universe and the impact of the protein structure initiative. *Proceedings of the National Academy of Sciences*, 111(10):3733–3738, 2014.

[46] Bisma S Khan and Muaz A Niazi. Network community detection: A review and visual survey. *arXiv preprint arXiv:1708.00977*, 2017.

[47] Andrew D King, Natasa Pržulj, and Igor Jurisica. Protein complex prediction via cost-based clustering. *Bioinformatics*, 20(17):3013–3020, 2004.

[48] Lars Kolb, Ziad Sehili, and Erhard Rahm. Iterative computation of connected graph components with mapreduce. *Datenbank-Spektrum*, 14(2): 107–117, 2014.

[49] Oleksii Kuchaiev, Marija Rašajski, Desmond J Higham, and Nataša Pržulj. Geometric de-noising of protein-protein interaction networks. *PLoS computational biology*, 5(8):e1000454, 2009.

[50] Roman A Laskowski, James D Watson, and Janet M Thornton. Profunc: a server for predicting protein function from 3d structure. *Nucleic acids research*, 33(suppl_2):W89–W93, 2005.

[51] Jooyoung Lee, Peter L Freddolino, and Yang Zhang. Ab initio protein structure prediction. In *From protein structure to function with bioinformatics*, pages 3–35. Springer, 2017.

[52] Sheng-An Lee, Cheng-hsiung Chan, Chi-Hung Tsai, Jin-Mei Lai, Feng-Sheng Wang, Cheng-Yan Kao, and Chi-Ying F Huang. Ortholog-based protein-protein interaction prediction and its application to inter-species interactions. *BMC bioinformatics*, 9(12):1–9, 2008.

[53] Christian M-R Lemer, Marianne J Rooman, and Shoshana J Wodak. Protein structure prediction by threading methods: evaluation of current techniques. *Proteins: Structure, Function, and Bioinformatics*, 23(3):337–355, 1995.

[54] Henry CM Leung, Qian Xiang, Siu-Ming Yiu, and Francis YL Chin. Predicting protein complexes from ppi data: a core-attachment approach. *Journal of Computational Biology*, 16(2):133–144, 2009.

[55] Xiaoli Li, Min Wu, Chee-Keong Kwoh, and See-Kiong Ng. Computational approaches for detecting protein complexes from protein interaction networks: a survey. *BMC genomics*, 11(1):1–19, 2010.

[56] Olivier Lichtarge, Henry R Bourne, and Fred E Cohen. An evolutionary trace method defines binding surfaces common to protein families. *Journal of molecular biology*, 257(2):342–358, 1996.

[57] Michael F Lin, Irwin Jungreis, and Manolis Kellis. Phylocsf: a comparative genomics method to distinguish protein coding and non-coding regions. *Bioinformatics*, 27(13):i275–i282, 2011.

[58] Guimei Liu, Jinyan Li, and Limsoon Wong. Assessing and predicting protein interactions using both local and global network topological metrics. In *Genome Informatics 2008: Genome Informatics Series Vol. 21*, pages 138–149. World Scientific, 2008.

[59] Quanzhong Liu, Jiangning Song, and Jinyan Li. Using contrast patterns between true complexes and random subgraphs in ppi networks to predict unknown protein complexes. *Scientific reports*, 6(1):1–15, 2016.

[60] Wei Liu, Liangyu Ma, Byeungwoo Jeon, Ling Chen, and Bolun Chen. A network hierarchy-based method for functional module detection in protein–protein interaction networks. *Journal of theoretical biology*, 455:26–38, 2018.

[61] Yaniv Loewenstein, Domenico Raimondo, Oliver C Redfern, James Watson, Dmitrij Frishman, Michal Linial, Christine Orengo, Janet Thornton, and Anna Tramontano. Protein function annotation by homology-based inference. *Genome biology*, 10(2):1–8, 2009.

[62] Phillip W. Lord, Robert D. Stevens, Andy Brass, and Carole A. Goble. Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation. *Bioinformatics*, 19(10):1275–1283, 2003.

[63] Edward M Marcotte, Matteo Pellegrini, Ho-Leung Ng, Danny W Rice, Todd O Yeates, and David Eisenberg. Detecting protein function and protein-protein interactions from genome sequences. *Science*, 285(5428): 751–753, 1999.

[64] Mark R Martzen, Stephen M McCraith, Sherry L Spinelli, Francy M Torres, Stanley Fields, Elizabeth J Grayhack, and Eric M Phizicky. A biochemical genomics approach for identifying genes by the activity of their products. *Science*, 286(5442):1153–1155, 1999.

[65] John Moult, Krzysztof Fidelis, Andriy Kryshtafovych, Torsten Schwede, and Anna Tramontano. Critical assessment of methods of protein structure prediction (casp)—round x. *Proteins: Structure, Function, and Bioinformatics*, 82:1–6, 2014.

[66] Kenta Nakai. Psort: a program for detecting the sorting signals of proteins and predicting their subcellular localization. *Trends Biochem. Sci*, 24(1): 34–35, 1999.

[67] Tamás Nepusz, Haiyuan Yu, and Alberto Paccanaro. Detecting overlapping protein complexes in protein-protein interaction networks. *Nature methods*, 9(5):471, 2012.

[68] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[69] Chandra Shekhar Pareek, Rafal Smoczynski, and Andrzej Tretyn. Sequencing technologies and genome sequencing. *Journal of applied genetics*, 52(4): 413–435, 2011.

[70] Florencio Pazos and Alfonso Valencia. In silico two-hybrid system for the selection of physically interacting protein pairs. *Proteins: Structure, Function, and Bioinformatics*, 47(2):219–227, 2002.

[71] Matteo Pellegrini, Edward M Marcotte, Michael J Thompson, David Eisenberg, and Todd O Yeates. Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. *Proceedings of the National Academy of Sciences*, 96(8):4285–4288, 1999.

[72] Xiaoqing Peng, Jianxin Wang, Wei Peng, Fang-Xiang Wu, and Yi Pan. Protein–protein interactions: detection, reliability assessment and applications. *Briefings in bioinformatics*, 18(5):798–819, 2017.

[73] Jose B Pereira-Leal, Anton J Enright, and Christos A Ouzounis. Detection of functional modules from protein interaction networks. *PROTEINS: Structure, Function, and Bioinformatics*, 54(1):49–57, 2004.

[74] Charles M Perou, Stefanie S Jeffrey, Matt Van De Rijn, Christian A Rees, Michael B Eisen, Douglas T Ross, Alexander Pergamenschikov, Cheryl F Williams, Shirley X Zhu, Jeffrey CF Lee, et al. Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. *Proceedings of the National Academy of Sciences*, 96(16):9212–9217, 1999.

[75] Sylvain Pitre, Md Alamgir, James R Green, Michel Dumontier, Frank Dehne, and Ashkan Golshani. Computational methods for predicting protein–protein interactions. *Protein–Protein Interaction*, pages 247–267, 2008.

[76] Shuye Pu, Jessica Wong, Brian Turner, Emerson Cho, and Shoshana J Wodak. Up-to-date catalogues of yeast protein complexes. *Nucleic acids research*, 37(3):825–831, 2009.

[77] Guimin Qin and Lin Gao. Spectral clustering for detecting protein complexes in protein–protein interaction (ppi) networks. *Mathematical and Computer Modelling*, 52(11-12):2066–2074, 2010.

[78] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

[79] Karthik Raman. Construction and analysis of protein–protein interaction networks. *Automated experimentation*, 2(1):1–11, 2010.

[80] Guillaume Rigaut, Anna Shevchenko, Berthold Rutz, Matthias Wilm, Matthias Mann, and Bertrand Séraphin. A generic protein purification method for protein complex characterization and proteome exploration. *Nature biotechnology*, 17(10):1030–1032, 1999.

[81] Alexander W Rives and Timothy Galitski. Modular organization of cellular networks. *Proceedings of the national Academy of sciences*, 100(3):1128–1133, 2003.

[82] Burkhard Rost, Predrag Radivojac, and Yana Bromberg. Protein function in precision medicine: deep understanding with machine learning. *FEBS letters*, 590(15):2327–2341, 2016.

[83] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[84] Manoj Pratim Samanta and Shoudan Liang. Predicting protein functions from redundancies in large-scale protein interaction networks. *Proceedings of the National Academy of Sciences*, 100(22):12579–12583, 2003.

[85] Benno Schwikowski, Peter Uetz, and Stanley Fields. A network of protein–protein interactions in yeast. *Nature biotechnology*, 18(12):1257–1261, 2000.

[86] Roded Sharan, Igor Ulitsky, and Ron Shamir. Network-based prediction of protein function. *Molecular systems biology*, 3(1):88, 2007.

[87] Peter HA Sneath, Robert R Sokal, et al. *Numerical taxonomy. The principles and practice of numerical classification.* 1973.

[88] Amy Hin Yan Tong, Becky Drees, Giuliano Nardelli, Gary D Bader, Barbara Brannetti, Luisa Castagnoli, Marie Evangelista, Silvia Ferracuti, Bryce Nelson, Serena Paoluzi, et al. A combined experimental and computational strategy to define protein interaction networks for peptide recognition modules. *Science*, 295(5553):321–324, 2002.

[89] Alfonso Valencia and Florencio Pazos. Computational methods for the prediction of protein interactions. *Current opinion in structural biology*, 12 (3):368–373, 2002.

[90] Stijn Marinus Van Dongen. *Graph clustering by flow simulation.* PhD thesis, 2000.

[91] Alexei Vazquez, Alessandro Flammini, Amos Maritan, and Alessandro Vespignani. Global protein function prediction from protein-protein interaction networks. *Nature biotechnology*, 21(6):697–700, 2003.

[92] Jianxin Wang, Min Li, Jianer Chen, and Yi Pan. A fast hierarchical clustering algorithm for functional modules discovery in protein interaction networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(3):607–620, 2010.

[93] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

[94] Cyrus A Wilson, Julia Kreychman, and Mark Gerstein. Assessing annotation transfer for genomics: quantifying the relations between protein sequence, structure and function through traditional and probabilistic scores. *Journal of molecular biology*, 297(1):233–249, 2000.

[95] Jerome Wojcik and Vincent Schächter. Protein-protein interaction map inference using interacting domain profile pairs. *Bioinformatics*, 17(suppl_1): S296–S305, 2001.

[96] Min Wu, Xiaoli Li, Chee-Keong Kwoh, and See-Kiong Ng. A core-attachment based method to detect protein complexes in ppi networks. *BMC bioinformatics*, 10(1):1–16, 2009.

[97] Jierui Xie, Boleslaw K Szymanski, and Xiaoming Liu. Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In *2011 ieee 11th international conference on data mining workshops*, pages 344–349. IEEE, 2011.

[98] Lei Xie and Philip E Bourne. Functional coverage of the human genome by existing structures, structural genomics targets, and homology models. *PLoS Comput Biol*, 1(3):e31, 2005.

[99] Evgeni M Zdobnov and Rolf Apweiler. Interproscan–an integration platform for the signature-recognition methods in interpro. *Bioinformatics*, 17(9): 847–848, 2001.

[100] Qiangfeng Cliff Zhang, Donald Petrey, Lei Deng, Li Qiang, Yu Shi, Chan Aye Thu, Brygida Bisikirska, Celine Lefebvre, Domenico Accili, Tony Hunter, et al. Structure-based prediction of protein–protein interactions on a genome-wide scale. *Nature*, 490(7421):556–560, 2012.

List of Figures

List of Tables