Analysis of Biological Networks by Graph Theory-based Methods(Dissertation_全文)

AUTHOR(S):

Li, Ruiming

KYOTO UNIVERSITY

DISSERTATION

Analysis of Biological Networks by Graph Theory-based Methods

生物情報ネットワークのグラフ理論に基づく解析法

Author:

Ruiming LI

Supervisor

Prof. AKUTSU Tatsuya

Abstract

As a major interdisciplinary area between informatics and life science, bioinformatics has achieved remarkable results in various fields. In this dissertation, we consider applying graph theory-based methods to solve two kinds of bioinformatics problems. One is the prediction of cancer genes by using weighted minimum feedback vertex sets (WMFVS), and another is the prediction of hot spot residues in protein complexes using densest subgraph-based (DS-based) methods.

In *Chapter 3*, we introduce our research in predicting cancer genes by WMFVS methods. Recently, many computational methods have been proposed to predict cancer genes. One typical kind of method is to find the differentially expressed genes between tumour and normal samples. However, there are also some genes, for example, 'dark' genes, that play important roles at the network level but are difficult to find by traditional differential gene expression analysis. In addition, network controllability methods, such as the MFVS method, have been used frequently in cancer gene prediction. However, the weights of vertices (or genes) are ignored in the traditional MFVS methods, leading to difficulty in finding the optimal solution because of the existence of many possible MFVSs. In this research, we developed a novel method, called WMFVS, which integrates the gene differential expression value with MFVS to select the maximum-weighted MFVS from all possible MFVSs in a protein interaction network. Our experimental results show that WMFVS achieves better performance than using traditional bio-data or network-data analyses alone. This method balances the advantage of differential gene expression analyses and network analyses, improves the low accuracy of differential gene expression analyses, and decreases the instability of pure network analyses. Furthermore, WMFVS can be easily applied to various kinds of networks, providing a useful framework for data analysis and prediction.

In *Chapter 4*, we introduce our research in predicting hot spot residues in protein

complexes by DS-based methods. Hot spots play an important role in protein binding analysis. The residue interaction network is a key point in hot spot prediction, and several graph theory-based methods have been proposed to detect hot spots. Although the existing methods can find some interesting residues by network analysis, the low recall has limited their abilities in finding more potential hot spots. In this study, we developed three graph theory-based methods to predict hot spots from only a single residue interaction network. We detect the important residues by finding subgraphs with high densities, i.e., high average degrees. Generally, a residue with a high degree in the residue interaction network implies a high binding possibility between protein chains. Thus, a subgraph with high density usually relates to binding sites that have a high rate of hot spots. By evaluating the results on 67 complexes from the SKEMPI database, our methods clearly outperform existing graph theory-based methods on recall and F-score, they provide useful approaches for analyzing bionetworks. In addition, the densest subgraph-based methods predict hot spots with only one residue interaction network, which is constructed from spatial atomic coordinate data to mitigate the shortage of data from wet-lab experiments.

Acknowledgements

I would like to appreciate my supervisor Prof. Akutsu's kind guidance and help during my doctoral program.

I also want to thank my parents and my friends who are supporting me in my daily life.

Contents

Figure contents

Table contents

Chapter 1

Introduction

1. Introduction

1.1. Background

As one of the branches of natural science, the research of biology has never been stopped along with human history. Biology helps us understand the living world, and it is essential for many fields such as agriculture, animal husbandry, medicine, and pharmacy. Traditional biology is based on macroscopic experiments and statistics according to fundamental units of life. Although traditional experimental methods have gotten remarkable achievements in various aspects of people's lives, in many cases it is hard to obtain the actual principle and obstructs the further development of biology. Fortunately, nowadays we can use microscopic experiments to research the essence of living beings. The study of genetics helps us orientation-breeding from bacteria to crops; the study of molecular biology helps us design drugs, and diagnose and treat disease; the study of immunology reveals our immune system, and also contributes to the research of cancer. There is no doubt that microbiology greatly increased the quality of our daily life.

With the development of modern biology, vast amounts of data emerge from laboratories over the world. Now there arises a demand for efficiently analyzing the existing big data, which gives birth to a new research field – bioinformatics. Bioinformatics aims at finding the relationship among various bio-data to explain the experimental results or predict unknown information. Many biology experiments tend to be notoriously expensive and time-consuming, while bioinformatics can predict the possible results of the experiments in advance, which accelerates the experimental cycle and saves precious resources. Recently, bioinformatics has been widely applied to assist wet experiments, especially in studying genomes and DNA sequencing. Another popular research field in bioinformatics is the analysis and prediction of protein-protein interaction (PPI). The study of PPI helps researchers understand the fundamental processes in living cells, and can also contribute to drug development and disease

treatment. Bioinformatics is now playing an important role in life sciences.

There are many different techniques applied in bioinformatics, such as pattern recognition, sequence analysis, image analysis, and data storing and management. With the prevalence of artificial intelligence, many machine learning (ML) methods, such as random forest, SVM, and deep learning, have been applied in bio-data analysis and prediction. The advantage of ML methods is that they can maximize the utilization of big data, and automatically find the potential association among bio-features. Various ML methods have achieved outstanding results in many bioinformatics issues. However, traditional ML-based methods also have some disadvantages in bioinformatics research, such as the difficulty in explaining the prediction results, the requirement of a large scale of training data, and the long periods and excessive computation resources expended in training models. Furthermore, in many fields, the data compose network structures, while it is hard to apply the topological information of the networks to general ML methods.

To fully make use of the network structures in bioinformatics, the graph theory-based methods are undoubtedly good choices. Compared to ML methods, the results of well-designed graph theory methods have high interpretability. Besides, graph theory methods do not require training data, which is an advantage in solving problems with insufficient experiment data. Also, unlike the huge models of ML methods, graph theory-based methods are usually light and fast, and they are useful tools for quick data checks or filters. As a popular field in computer science, there exist a plenty of mature graph theory algorithms for network analysis. These methods could help researchers flexibly deal with bioinformatics problems.

1.2. Related works

There exist many graph theory-based methods for solving various bioinformatics problems. Wei et al. [1] study the biological features of biological networks in terms of eccentric topological indices computation. The conclusions in this paper illustrate that bioengineering has promising application prospects. Jack et al. [2] use tools from graph

theory to define an Atlas classification scheme for automatically categorizing certain protein substructures. Applying a graph theory-based Atlas classification scheme gives the Atlas of Coiled Coils, a fully automated, updated overview of extant coiled coils. Xingqi et al. [3] construct novel mathematical descriptors based on graph theory to determine the DNA sequence similarity. This new approach measures similarity based on both ordering and frequency of nucleotides so that much more information is involved compared to traditional methods. Néli et al. [4] apply the complex networks theory to map groups of functionally related residues in residue co-evolutionary networks, and successfully detected several specificity determinant sets and functional motifs. Ertan et al. [5] introduce a graph theory-based classification model for diagnostic purposes that can be easily adapted for different neurological diseases, the results show that the graph-based measures computed on brain connectivity networks might help to improve the diagnostic capability of *in-silico* methods. Matej et al. [6] discuss an approach developed for exploiting the local elementary movements of evolution to study complex networks in terms of shared common embedding and, consequently, shared fractal properties. This approach can be useful for the analysis of lung cancer DNA sequences and their properties by using the concepts of graph theory and fractal geometry. Jacob et al. [7] present a methodology for graph-based enumeration of surfaces and unique chemical adsorption structures bonded to those surfaces. These techniques are useful for generating a wide variety of structures used in computational surface science and heterogeneous catalysis, and are also key to facilitating an informatics approach to the high-throughput search for more effective catalysts. Spyridon et al. [8] present a novel graph-based methodology for the development of structural and functional brain graphs. Graph theory-based analysis has been applied with great success in studying the brain's connectivity, organization, and dynamics. Roy et al. [9] propose a methodological parallel between Quality Threshold (QT) clustering and Maximum Clique algorithm, which significantly contributes to reaching a very affordable algorithm compared to the few implementations of QT for molecular dynamics available in the literature. Mahnaz et al [10] use the contact map of

a protein to construct a graph, and then analyze a protein's structure without e three-dimensional (3D) coordinates data.

1.3. Contribution

In this dissertation, we introduce two novel bioinformatics researches based on graph theory algorithms.

The first research, named WMFVS, combines the gene differential expression data with minimum feedback vertex set (MFVS) from a human protein interaction network to predict cancer genes. The proposed method improved the low accuracy of traditional differential expression-based prediction methods, and was much more stable than pure MFVS methods. The results show that our WMFVS methods can successfully predict cancer genes, and these methods can be easily applied to other bioinformatics network analysis problems.

In the second research, we apply the densest subgraph-based method to predict hot spot residues in protein complexes. We propose three different methods, each of them has different advantage in precision or recall, and all these three methods have the ability to detect possible hot spot residues just from the 3-dimensional data of complexes. Our methods provide new models in bioinformatics network analysis.

1.4. Organization

In Chapter 2, we briefly introduce several basic background knowledge about graph theory and linear programming (LP), together with the definitions of some bioinformatics problems.

In Chapter 3, we introduce our research in the cancer gene prediction problem. In this research, we combined the traditional gene differential expression-based methods with a graph-based method, and two major methods are proposed and applied to the human protein interaction network. The results are evaluated by several independent cancer gene data sets.

In Chapter 4, we introduce our research in the hot spot residue prediction problem. Three different densest subgraph-based methods are developed and applied to 341 + 27 protein complexes, and 67 prediction results are evaluated by existing hot spot data sets. Furthermore, the results are compared with some other graph theory-based methods.

In Chapter 5, we give a conclusion to these two studies, together with discussions of future works.

Chapter 2

Preliminaries

2. Preliminaries

2.1. Graph structure

Graph is a data structure composed by vertices and edges (or arcs) between the vertices. Each vertex usually refers a sample in real world, such as a gene or a residue, and thus a vertex has its own features like ID or weight. An edge between two vertices indicates there exists a relationship between the vertices, for example protein interaction or residue interaction. Graph can be directed or undirected. In a general directed graph, all the edges have directions, i.e. the relationships between vertices have directions, while in undirected graph, the relationships have no direction.

Figure 2.1 shows a simple example of a directed graph.

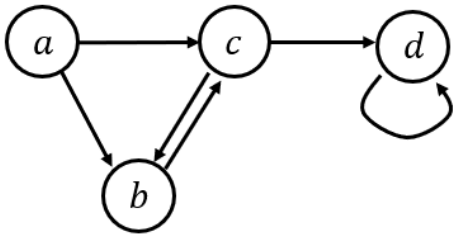


Figure 2.1 An example of a directed graph.

Here we use $G_1 = (V_1, E_1)$ to denote this graph, where $V_1 = \{a, b, c, d\}$ is the set of vertices, and $E_1 = \{(a, b), (a, c), (b, c), (c, b), (c, d), (d, d)\}$ is the set of edges. Note that:

- each edge in E_1 has a fixed direction;
- two reverse edges may exist simultaneously between two vertices;
- an edge can start and end at the same vertex (self-loop).

Besides, we can also use an adjacency matrix M_1 to represent G_1 as Figure 2.2.

	a	b	c	d
a	0	1	1	0
b	0	0	1	0
c	0	1	0	1
d	0	0	0	1

Figure 2.2 The adjacency matrix M_1 of G_1.

In the adjacency matrix, $M_1[i,j] = 1$ represents there exists an edge $(i,j) \in E_1$, and vice versa. The number of 1's in M_1 represents the number of edges in G_1, the number of 1's in a row represents the out-degree (number of edges starting from the vertex) of a vertex, and the number of 1's in a column represents the in-degree (number of edges end at the vertex) of a vertex.

Figure 2.3 shows an undirected graph.

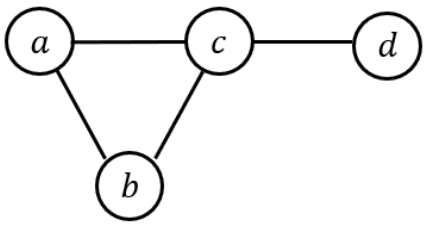


Figure 2.3 An example of an undirected graph.

We use $G_2 = (V_2, E_2)$ to denote the graph above, where $V_2 = \{a, b, c, d\}$ and $E_2 = \{(a,b), (a,c), (b,c), (c,d)\}$. Note that:

- edges in E_2 have no direction;
- in this dissertation, unless otherwise specified, we only discuss the simple graph (or strict graph) for the undirected graph, i.e. at most one edge exists between two vertices, and there is no self-loop in the undirected graph.

We can also use an adjacency matrix M_2 to represent G_2 as Figure 2.4.

	a	b	c	d
a	0	1	1	0
b	1	0	1	0
c	1	1	0	1
d	0	0	1	0

Figure 2.4 The adjacency matrix M_2 of G_2.

Generally, the adjacency matrix of an undirected graph is a symmetric Matrices, and the value in each cell is Boolean and refers to the existence of the related edge (in some cases, we can modify the value in a cell to some real number to represent the weight of an edge). The number of 1's in a row equal to the number of edges connected to the corresponding vertex, we use "degree" to denote this value, i.e. the number of neighbors that a vertex connects to.

The graphs discussed in *Chapter 3* are directed graphs (protein interaction network), while the graphs discussed in *Chapter 4* are undirected graphs (residue interaction network).

2.2. Feedback vertex set

Given a directed graph $G = (V, E)$, a feedback vertex set (FVS) $S \subseteq V$ is a set of vertices, whose removal leaves the remaining network acyclic, i.e. $G' = (V', E')$ has no circle, where $V' = V - S$ and $E' = \{(i, j) \in E | i \in V', j \in V'\}$. If an FVS S has the minimum size among all possible FVS's, S is called a minimum FVS, or MFVS for short. Figure 2.5 shows four simple examples of FVS and MFVS.

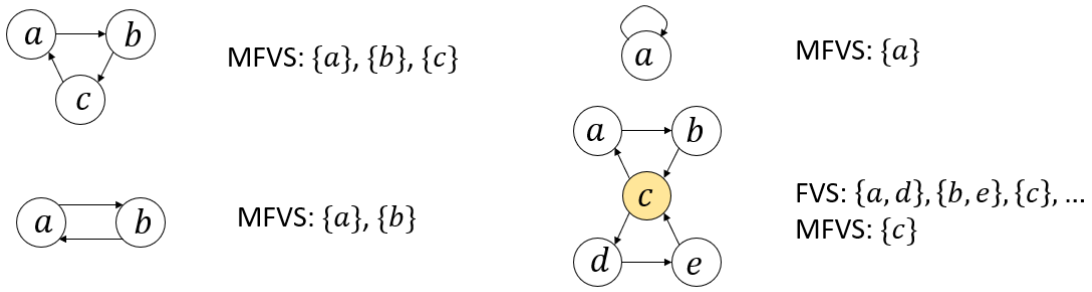


Figure 2.5 Some examples of FVS and MFVS. In the right-bottom graph, only the set $\{c\}$ is an MFVS.

A given graph may have multiple MFVSs. Finding an arbitrary MFVS from a given MFVS is proved to be NP-hard. An integer linear programming-based method is introduced in *Chapter 3* to solve the MFVS problem.

The MFVS based-methods are widely used in operating systems, database systems, and also bioinformatics. In *Chapter 3*, we will introduce a variant MFVS based-method to find cancer genes from a human protein interaction network, which integrates the weights of vertices from the gene differential expression data.

2.3. Strongly connected component

Given a directed labeled graph $G = (V, E)$, a subgraph $S = (V', E')$ is a strongly connected subgraph (scs), if for any pair of vertices (a, b) in V', there always exists a path from a to b. If a scs S is not included by any other scs, then S is a strongly connected component (scc) of G. Using Gabow's algorithm [11] can find all the scc's of a given graph $G = (V, E)$ in $O(|V| + |E|)$ time.

For any two scs's $S_1 = (V_1, E_1)$ and $S_2 = (V_2, E_2)$, if there exist two edges $(v_1, u_1) \in E$ and $(u_2, v_2) \in E$ that $v_i \in V_1$ and $u_i \in V_2$, then obviously $S_3 = S_1 \cup S_2$ is still an scs. Thus if S_1 and S_2 are two scc's and an edge from S_1 to S_2 exists, then no edge from S_2 to S_1 exists. Furthermore, we have the following proposition.

Proposition 1. *Any edge between two different scc's is not included in any circle.*

Proof. Assume there exists an edge (a, b) between two different scc's $a \in S_1$ and $b \in$

S_2, and (a, b) is included in a circle c. Obviously, path c composes an scs C. Then $S_3 = S_1 \cup C$ is still an scs and contains at least one vertex $b \notin S_1$, i.e. $S_3 \supset S_1$, which contradicts the assumption that S_1 is an scc. Thus, such a circle c does not exist.

■

We will use Proposition 1 to compress graphs in *Chapter 3*.

2.4. Densest subgraph

Given an undirected graph $G = (V, E)$, let $V' \subseteq V$ and $S = (V', E')$ be the subgraph of G induced by V'. Then the density of S is defined by $\rho(S) = \frac{|E'|}{|V'|}$. For simple graphs, $\rho(S)$ is also proportional to the average degree of vertices, i.e. proportional to the connectivity of a graph. If a subgraph S has the maximum density among all possible subgraphs of G, then S is a densest subgraph of G, and the density of G, denoted by $D(G)$, equals to the density of S, i.e. $D(G) = \rho(S)$.

A given graph may have multiple densest subgraphs. The intersection (if not empty set) or union of several densest subgraphs is still a densest subgraph. See Figure 2.6.

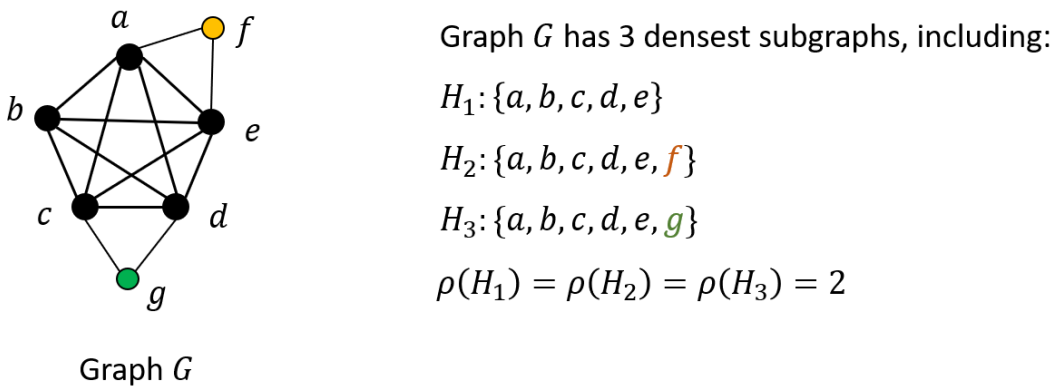


Figure 2.6 An example of densest subgraphs. In this example, $\boldsymbol{G}$ has multiple densest subgraphs. The density of $\boldsymbol{G}$ is 2.

Based on the definition of densest subgraph, we can further define the minimal and maximal densest subgraphs. Let S be a densest subgraph of G, if there exists no densest

subgraph S' that $S' \subset S$, then S is a minimal densest subgraph; if there exists no densest subgraph S' that $S \subset S'$, then S is a maximal densest subgraph. In the example in Figure 2.6, H_1 is a minimal densest subgraph, and the whole graph itself is a maximal densest subgraph.

A given graph may have multiple minimal densest subgraphs, but only exactly one maximal densest subgraph exists.

2.5. Linear Programming

Linear programming (LP) is widely used technique in optimization field, i.e. finding a feasible assignment (if exists) of input variables to maximize or minimize the objective function. When all the variables in a linear program are real values, all polynomial-size linear programs can be solved in polynomial time. However, when at least one variable are forcibly constrained to be integers, this linear program is said to be an integer linear program (ILP) and this problem is generally NP-hard. A typical LP (or ILP) can be expressed in canonical form like:

Find a vector $\boldsymbol{x}$ that

maximize	$\boldsymbol{c}^T\boldsymbol{x}$	(objective function)
subject to	$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$	(constraints)
and	$\boldsymbol{x} \geq \boldsymbol{0}$	(constraints)

Here $\boldsymbol{x}$ is the set of variables to be determined, $\boldsymbol{A}$ (matrix), $\boldsymbol{b}$ (vector), and $\boldsymbol{c}$ (vector) are fixed parameters to constraint the value of feasible $\boldsymbol{x}$.

In this dissertation, LP and ILP-based techniques are the core of our WMFVS and densest subgraph-related methods. All LP or ILP-based models are implemented by the Gurobi solver [12].

2.6. Cancer genes

The cancer gene is a kind of gene whose abnormal expression may lead to cancer diseases. There are two subclasses of cancer genes, one is the oncogene (positive growth

regulators), whose overexpression may lead to cancer, and the other one is the tumour suppressor gene (TSG) (negative growth regulators), whose insufficient expression may lead to cancer. Figure 2.7 shows the relationship between cancer genes and cancer diseases.

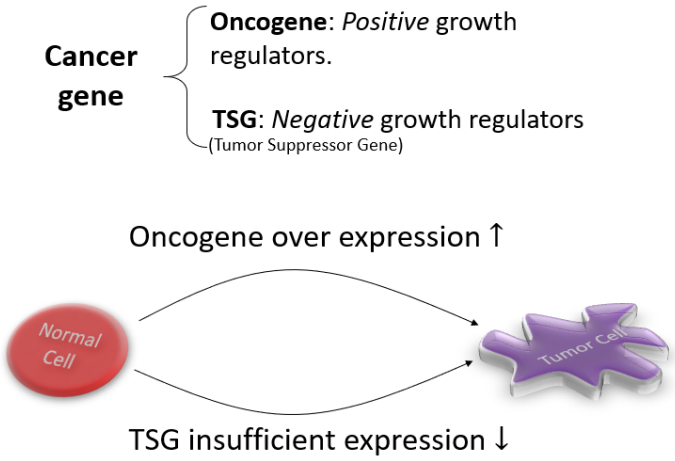


Figure 2.7 Oncogene and TSG.

The traditional way to detect cancer genes is based on the differential gene expression value between normal cells and cancer (tumor) cells. Generally, higher differential expression value represents high possibility of cancer gene. See Figure 2.8.

Normal samples

	Gene1	Gene2
N1	1.1	1.0
N2	1.0	0.8
N3	0.8	0.9
N4	0.3	1.1

Tumor samples

	Gene1	Gene2
T1	0.1	0.9
T2	0.2	1.0
T3	0.1	1.2
T4	0.1	0.8

High differential value

Low differential value

Figure 2.8 An example of cancer gene and normal gene. Compared to Gene2, Gene1 is more like a cancer gene.

2.7. Hot spot residues

In protein complexes, two or more proteins interact with others by binding surfaces. However, not all residues in the binding surfaces are related to protein-protein interaction, only a small portion of interface residues, called hot spots, contribute the majority of the binding energy. Figure 2.9 shows an example of a protein complex and hot spot residues.

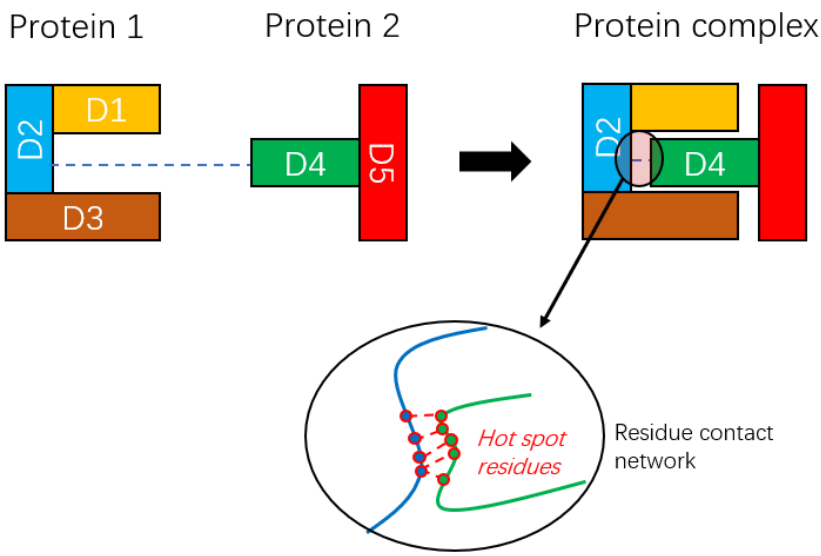


Figure 2.9 An example of the hot spot residues.

In the example in Figure 2.9, a protein complex is composed of two proteins, these two proteins are composed of functional domains $\{D1, D2, D3\}$ and $\{D4, D5\}$, respectively. There exists one binding surface between $D2$ and $D4$, while only the red circled residues (hot spots) contribute to the binding energy of these two proteins.

The traditional way to detect hot spots is based on residue mutation methods. If a mutation of a residue (usually mutated to alanine, which hardly interacts with other residues but mimics the wild-type secondary structure) in a protein-protein interface changes the binding energy of the protein to its binding partner substantially (change of binding energy $\Delta\Delta G \geq 2.0$ kcal/mol), then this residue is defined as a hotspot residue. Based on the alanine scanning experiments data, the Alanine Scanning Energetics Database (ASEdb) [13] is built for searching hot spot residues. However, since the related

mutation experiments are time-consuming and laborious, the experimental approved hot spots are sparse, thus there arises a need for computational methods for hot spot prediction.

2.8. Protein Data Bank

Research Collaboratory for Structural Bioinformatics Protein Data Bank (RCSB PDB, or usually PDB for short) [14] is one of the most famous experimentally-determined protein 3D structures data sets. It contains atom 3D spatial coordinate data from around 200 thousand protein structures and has a stable and convenient searching system. In Chapter 4, we will use the spatial data from PDB to construct the residues interaction networks.

Chapter 3

Weighted minimum feedback vertex sets and implementation in human cancer genes detection

3. Weighted minimum feedback vertex sets and implementation in human cancer genes detection

3.1. Background

Cancer is a genetic disease, but not all genes are related to cancer. By almost universal consensus, cancer is now viewed as resulting from changes in some key regulatory genes [15]. At present, researchers have defined several kinds of cancer-related gene sets. One widely used kind of gene set is that of cancer driver genes, which are defined as genes whose mutations increase net cell growth under the specific micro-environmental conditions that exist in the cell in vivo. This kind of gene can be predicted by finding 'significantly mutated genes', whose mutation rates are significantly higher than the presumed background somatic mutation rate [16-18]. However, since it is difficult to construct a reliable background mutation model [19], selecting gold-standard driver genes by frequency-based methods is difficult. Another kind of cancer-related genes are so-called 'cancer genes', including oncogenes, which function as positive growth regulators, and TSGs, which function as negative growth regulators. These genes are directly related to the phenotypes of tumour and normal genes and can be predicted by differential gene expression analyses. However, some 'dark' genes play important roles at the network level but are generally ignored by traditional differential gene expression analyses [20, 21].

By using graph theory algorithms, we can find critical vertices to control a network. For example, [22] developed a feedback-based framework that provides realizable node overrides that steer a system towards one of its natural long-term dynamic behaviours; [23] provided a rational criterion for selecting key molecules to control a system with a FVS; [24] proposed a network control strategy to find driver mutations that drive a regulation network from the normal state to a disease state; [25] considered applying

MFVS to real biologically directed complex networks and found essential proteins in both Drosophila melanogaster and Homo sapiens organisms; and [26] proposed an MFVS-based framework for controlling multilayer networked structures.

Given a directed network, an FVS is a set of vertices whose removal leaves the remaining network acyclic. The MFVS is a kind of FVS that has the minimum size among all possible FVSs. The MFVS problem has been proven to be NP-complete [27]. There already exist many algorithms for solving the MFVS problem, including approximation algorithms [28], randomized algorithms [29], parameterized algorithms [30] and exact algorithms [31, 32].

Generally, a network can have multiple MFVSs. Traditional MFVS algorithms ignore the differences among possible MFVSs, and the output is usually uncertain. In the worst case, $O(2^{|v|/2})$ MFVSs may exist in a graph [33]. This uncertainty leads to the instability of network analysis methods in practice. However, in reality, vertices should have different weights, for example, the importance of different genes should be distinguished. Based on this consideration, to find the best output from multiple MFVSs, in this paper, we consider a variation of the MFVS problem, i.e., each vertex is assigned a weight, and the output is the maximum total weighted MFVS. The assigned weight should reflect the significance of the corresponding vertex, which may involve some biological data from other studies (for example, in our experiments, we utilize the differential expression value to compute the weights). We define this problem as a WMFVS problem.

To solve the WMFVS problem, we modified an exact algorithm from [32], which first compresses the original graph [34, 35] to reduce the number of vertices and arcs and then utilizes an integer linear programming (ILP) method for the compressed graph. Our WMFVS method can be roughly separated into three parts, i.e., graph compression, MFVS size determination and output optimization. The first two parts use the same idea as [32], and the third part uses the modified ILP method to select the maximum weighted MFVS.

Furthermore, we consider a variation of the WMFVS method that pays more attention

to the total weight of an FVS than to its size; i.e., it finds the maximum-weighted FVS. We call this method WFVS. In the next sections, we can see that WMFVS has a higher precision than WFVS, while WFVS has an advantage in recall.

3.2. Methods

3.2.1. Graph compression

In biological networks, a network usually contains tens of thousands of vertices and hundreds of thousands of arcs. In many cases, processing a large network is not practical because of the NP-hardness of the MFVS problem [27]. Generally, we can compress the original graph to a simpler graph that maintains (or can restore) the size of the MFVS of the original graph.

In the following sections, we define $v.suc$ and $v.pre$ as the sets of successors and predecessors of vertex v, respectively. Let v_i be a vertex in a network S. Consider the following three cases [34]:

C1. $v_i \in v_i.suc$, i.e., v_i has a self-loop; then, v_i should be in all FVSs, otherwise the self-loop cannot be removed.

C2. $v_i.suc = \emptyset$ (or $v_i.pre = \emptyset$); then, v_i is not in any MFVS, since it is not in any cycle.

C3. $|v_i.suc| = 1$ (or $|v_i.pre| = 1$); let v_j be the only successor (or predecessor, respectively) of v_i; then, any cycle containing v_i also contains v_j.

For C1, we use a temporary list ΔM to record v_i; we add v_i to ΔM and remove v_i and all its incoming and outgoing arcs from the graph. We use $remove(v_i)$ to denote this removing process. See Figure 3.1.

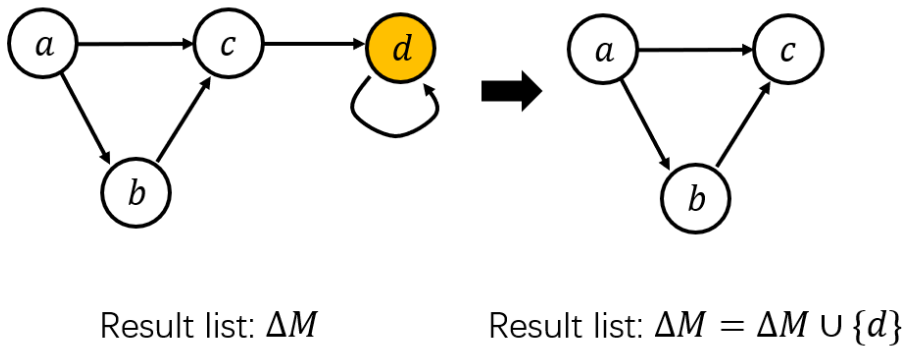


Figure 3.1 Example of C1. Since $\boldsymbol{d}$ has a self-loop, it is added to $\boldsymbol{\Delta M}$ and removed from the graph.

For C2, since v_i is not in any MFVS, we can safely use $remove(v_i)$ without any change to the possible MFVSs. See Figure 3.2.

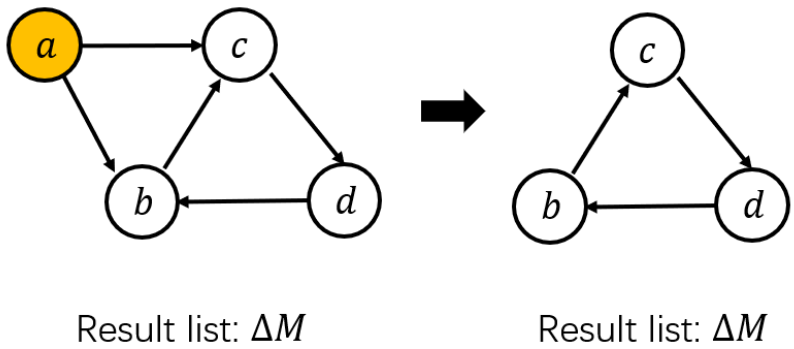


Figure 3.2 Example of C2. Since $\boldsymbol{a}$ has no successor, $\boldsymbol{a}$ is not in any MFVS, and thus it can be safely removed from the graph.

For C3, assume v_i is in some cycle c. If we attempt to break c by removing v_i, then it is equally good (sometimes better) to remove v_j rather than v_i. Here, we connect all predecessors of v_i to all its successors and then use $remove(v_i)$. We denote this connecting and removing operation by $ignore(v_i, S)$, where S is the current graph to which v_i belongs. The procedure is as follows:

Procedure *ignore*(*v*, *S*)

```
for v_i ∈ v.pre:
  for v_j ∈ v.suc:
    if (v_i, v_j) ∉ S.E:
      S.E := S.E ∪ {(v_i, v_j)}
remove(v)
```

Figure 3.3 shows an example of C3.

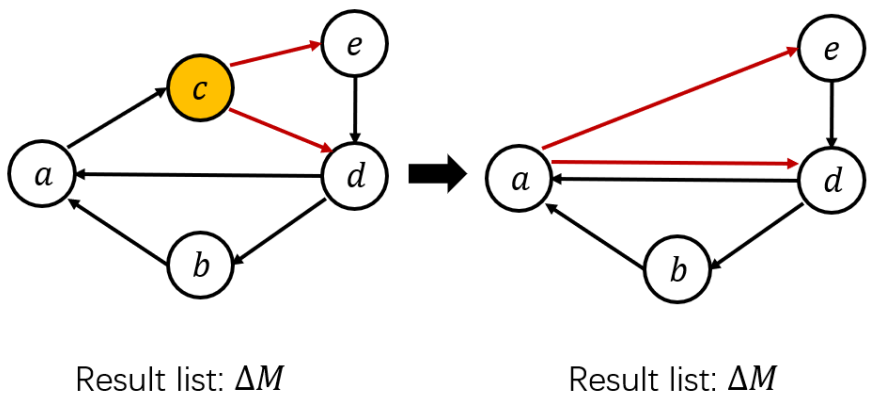


Figure 3.3 An example of C3.

In the example in Figure 3.3, c has only one successor a, then any circle including c must contain a (since it must contain arc (a, c)), thus removing a will break all the circles that include c, while a may be included in some other circles. In this case, we apply the *ignore* operation, which removes c but keeps all the circles that originally contain c for further analysis.

In the above procedure, v is a vertex in graph S, and $S.E$ is the arc set of graphs S. Then we have the following procedure to compress a graph S:

Procedure *compress_vertex*(S)*:*

```
ΔM := ∅
for v_i in S.V:
  if v_i ∈ v_i.suc:
    ΔM := ΔM ∪ {v_i};
    remove(v_i)
  else if |v_i.suc| == 0 or |v_i.pre| == 0:
    remove(v_i)
  else if |v_i.suc| == 1 or |v_i.pre| == 1:
    ignore(v_i, S)
return ΔM
```

We repeat this procedure until S cannot be modified. Furthermore, we use the strongly connected components (scc's) [32, 35] to reduce the arcs. Since an arc between two scc's is not in any cycle (by Proposition 1), the deletion of these arcs will not change any MFVSs. See Figure 3.4.

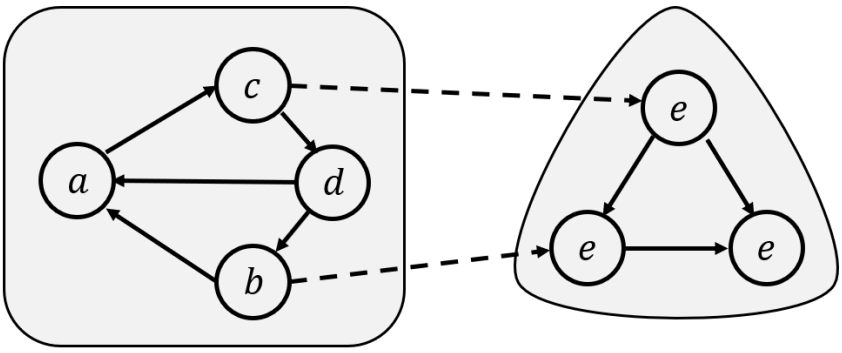


Figure 3.4 The arcs between scc's are not in any circles.

We use *compress_scc*(S) to denote the operation that removes all arcs between two different scc's in S. The whole graph compressing procedure is as follows:

Procedure *compress_graph*(S)*:*

```
ΔM := ∅
do {
  compress_scc(S)
  ΔM := ΔM ∪ compress_vertex(S)
} while S is modified and S.V ≠ ∅
return ΔM
```

The returned ΔM contains the vertices that are always in any MFVS, and the union of ΔM and any MFVS of the compressed graph will be an MFVS of the original graph.

Note that not all MFVSs of the original graph can be obtained from the above method. Some MFVSs are lost in the ignore operation, while in a weighted MFVS problem, the lost MFVSs may have the maximum weight. For the weighted case, we modify the ignore operation to consider the weights of vertices (only for positive-weighted cases). The following method ensures that the maximum-weight MFVS (the WMFVS) will not be lost:

Procedure *ignore_w*(v, S)*:*

```
if |v.suc| == 1:
  let v' be the only successor of v
  if v.w < v'.w:
    ignore(v, S)
else if |v.pre| == 1:
  let v' be the only predecessor of v
  if v.w < v'.w:
    ignore(v, S)
```

where $v.w$ denotes the weight of vertex v.

Proposition 2. *When the weights of vertices are positive, the ignored vertices in procedure ignore_w are not in any WMFVS.*

Proof: Assume $v.pre = \{v'\}$ and $v.w < v'.w$, and v belongs to a WMFVS M. Then $v' \notin M$, otherwise $M' := M - \{v\}$ is still an FVS, which has less vertices than an MFVS. Now consider $M'' := (M - \{v\}) \cup \{v'\}$. It is obvious that M'' is an MFVS. Since $v'.w > v.w$, we have $\Sigma_{v_i \in M} v_i.w < \Sigma_{v_j \in M''} v_j.w$. Thus M cannot be a WMFVS, i.e. if v has only one processor and the weight of v is less than the processor, then v dose not belong to any WMFVS. The proof is similar when v has only one successor and the weight of v is less than the successor.

■

3.2.2. ILP formulation for MFVS and WMFVS

After the compressing procedure, if the compressed graph is not empty, we can use an ILP method [32] to solve the remaining MFVS problem. For each remaining vertex v_i, we add two parameters x_i (Boolean) and k_i (integer), where x_i denotes whether v_i is included in the output MFVS result and k_i is a temporary parameter used in the ILP. The ILP formulation is as follows:

ILP1:

$$\textbf{Minimize} \quad \sum x_i$$

$$\textbf{Subject to} \quad k_i - k_j + nx_i \geq 1 \qquad \forall(v_i, v_j) \in E$$

$$0 \leq k_i \leq n - 1$$

where E is the arc set of the remaining graph. These constraints ensure that the selected vertices compose an FVS of S, while the objective function means that the selected FVS has a minimum size, i.e., it is an MFVS.

Now we consider the weighted case of the MFVS problem. Given a graph S, where each vertex $v_i \in S.V$ has a weight $v_i.w$ (in what follows, we use w_i to denote $v_i.w$ if there is no ambiguity), the WMFVS problem is to find an MFVS of S that has the maximum total weight. Assuming we already know the size s of the MFVS (by ILP1 or

some estimation method such as that of [36] or [37]), the following formulation optimizes the selected MFVS as a WMFVS:

ILP2:

$$\textbf{Maximize} \quad \sum w_i x_i$$

$$\textbf{Subject to} \quad \sum x_i = s$$

$$k_i - k_j + n x_i \geq 1 \qquad \forall (v_i, v_j) \in E$$

$$0 \leq k_i \leq n - 1$$

The constraint $\sum x_i = s$ ensures that the selected FVS is an MFVS, while the objective function selects the maximum-weight MFVS among all possible MFVSs.

3.2.3. Maximum-weight FVS

In the WMFVS problem, we first restrict the size of the FVS to be minimal and then select the maximum-weight MFVS as the objective. However, sometimes the weight may be more important than the size of an FVS. As an example, in Figure 3.5, the WMFVS is $\{b\}$, which has a total weight of -20. If we do not restrict the minimum size of the set, the FVS $\{a, c\}$, which has weight -4, seems better.

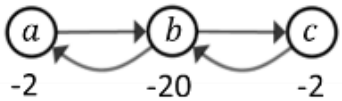


Figure 3.5 A simple example of FVS. In this case, the total weight may be more important than the size of an FVS.

Here we define a variant of the WMFVS problem, which ignores the exact size of the output vertex set, as follows: Given a graph S, where each vertex $v_i \in S.V$ has a weight $v_i.w$ (or w_i), the weighted FVS (WFVS) problem is to find an FVS of S that has the maximum total weight. We can simply use a similar ILP as ILP2 to solve the WFVS problem.

ILP3:

Maximize $\sum w_i x_i$

Subject to $k_i - k_j + nx_i \geq 1 \qquad \forall(v_i, v_j) \in E$

$0 \leq k_i \leq n - 1$

However, simply removing the constraint $\sum x_i = s$ may lead to a trivial solution when the weights of the vertices are positive, since the set of all vertices will always be a WFVS. Here we consider two methods to avoid the trivial solution:

1. Modify all weights to be negative. Assume the maximum weight of the vertices is w_m; then, for each weight w_i , modify it to $w_i \coloneqq w_i - w_m - \delta$. Here, δ is a small positive constant to ensure that all weights are negative. The ILP is the same as ILP3.
2. Reverse the weights to penalty values. We can simply do this by taking the inverse of each w_i, i.e.,

$$p_i = \begin{cases} \frac{1}{w_i}, \text{if } w_i \neq 0 \\ \infty, \text{if } w_i = 0 \end{cases}$$

Then, modify the ILP3 formula as follows:

ILP3':

Minimize $\sum p_i x_i$

Subject to $k_i - k_j + nx_i \geq 1 \qquad \forall(v_i, v_j) \in E$

$0 \leq k_i \leq n - 1$

In our research, we examined both ways of calculating the weights in the WFVS method. We found that the first modification is more unstable when running the ILP process, i.e., more obviously wrong ILP results appeared. Thus, we chose to use the second method to compute the weights in the WFVS method; i.e., we reversed the weights to be penalty values, which are always positive values.

In the second method, we need to avoid the 'division by zero' error. To this end, we used the simple heuristic formula below.

Let l be a large number (in our program, we used 65536); then, the penalty is calculated by:

$$p_i = \begin{cases} \frac{1}{w_i}, \text{if } w_i \geq \frac{1}{l} \\ l \ , \text{if } w_i < \frac{1}{l} \end{cases}$$

3.3. Results

3.3.1. Data sets

In this study, we used the directed human protein interaction network [38] for the analyses; it contains 6338 genes (vertices) and 34814 directed interactions (arcs). To evaluate the relative prediction accuracies for cancer genes between our methods and existing methods, we collected cancer-related gene sets from five public databases: ONGene [39], TSGene [40], CGC [41], NCG [42] and MSigDB C6 [43]. Since not all genes from the data sets are contained in the directed human protein interaction network, we filtered the common genes in both a certain data set and the network. The sizes of these data sets are shown in Table 3.1.

Table 3.1 Size of each data set and the number of genes contained in the network (common genes).

	ONGene	TSGene	CGC	NCG	MSigDB
Number of genes	803	1217	723	2372	10962
Common genes	490	641	525	1210	4184

In the rest parts, when we calculate the recall of various methods, we consider only the size of the common gene sets.

3.3.2. Weight definition

To define the weights of genes, we first downloaded the RNA-seq data from TCGA [44], which contains gene expression data from 1102 breast tumour samples and 113 normal

samples. Next, the counts of level 3 RNASeqV2 data were processed and transformed before being used for further analysis [45]. Specifically, we used the fold change (FC) value (with the binary logarithm and absolute value) between tumour and normal samples as the weight of each vertex (gene). For a specific gene v, its weight is calculated by the following formula:

$$v.w = \left| \frac{\sum_{i=1}^{n} log_2(T_i)}{n} - \frac{\sum_{j=1}^{m} log_2(N_j)}{m} \right|$$

where T_i is the expression value of tumour sample i, N_j is the expression value of a normal sample j, and n and m are the numbers of tumour and normal samples, respectively. Intuitively, a high FC value corresponds to a high possibility of a cancer gene. Thus, it is reasonable to use the FC values as the weights of genes.

For the genes that appear in the network but have no expression values in the TCGA data (only 143 genes, 2.3% of the network size; these are called weight-loss genes), we gave them default weights of 0 rather than ignoring them; thus, if such a gene is essential at the topological level, it has the potential to be selected as a cancer gene, which may counteract the disadvantage of the traditional differential expression-based methods in dark gene-revealing and missing-data situations. Finally, all 6338 genes in the graph were weighted. The topological structure of the graph remained the same as in the original protein interaction network.

3.3.3. Experiments and evaluation

The whole experiment process is shown in Figure 3.6.

First, we analyzed the directed human protein interaction network with traditional MFVSs and obtained a set of 463 vertices. Then, we used our WMFVS method on the same network (the weights were derived from the FC values). We also used the inverses of the weights as the penalty values and applied them to our WFVS method.

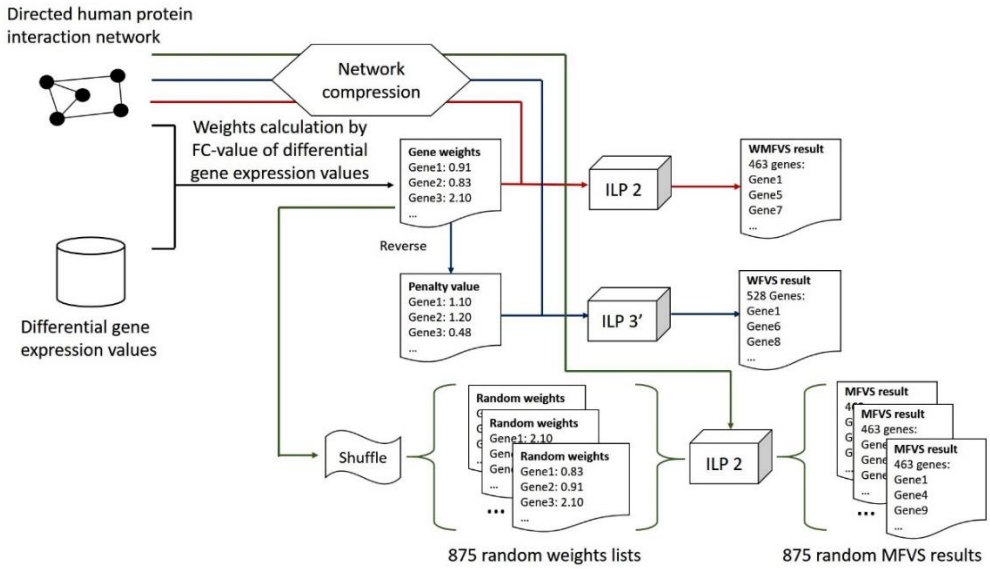


Figure 3.6 The experiment flowchart. The red, blue and green lines correspond to the WMFVS, WFVS and random MFVS pipelines, respectively.

Because of the non-uniqueness of the MFVS method, it is not a general evaluation if we consider only one MFVS result. Therefore, we calculated a set of random MFVSs by applying the WMFVS method with randomly shuffled gene weights. First, we planned to compute 1000 random MFVSs for analysis. However, since the Gurobi optimizer (version 8.1.0) does not always output a real optimal solution (e.g., even when we restrict the size of the output to be exactly 463, which is the size of the MFVS, sometimes the sizes of the output are smaller than 463), we filtered the obviously incorrect results and verified all the other outputs as MFVSs. Finally, we obtained 875 approved random MFVSs (since some MFVSs may be lost in the *ignore_w* operation and the MFVSs are not distributed uniformly, not all possible MFVSs have the same possibility of selection).

The WMFVS and WFVS result data can be found in the supplementary data. The random MFVS data are placed in https://github.com/lrming1993/WMFVS_codes.

To evaluate the results of these three methods, we first checked the graph-level results (see Table 3.2).

Table 3.2 The graph-level results of each method.

	Output Size	Run Time (second)	Sum Weight	Average Weight of each vertex
MFVS	463	4.0	319.5	0.69
WMFVS	463	35.9	379.3	0.82
WFVS	528	23.6	496.4	0.94

The run time of MFVS is due to the use of the traditional non-weighted MFVS method. The sum weight of MFVS uses the average value from 875 randomly weighted WMFVSs.

As we expected, the WMFVS method obtained a better total weight than the traditional MFVS. However, the result of WMFVS is not always better than that of MFVS. The total weight of the output of the traditional MFVS method is arbitrary (the output is related to the graph structure but has no relevance to the vertex weights), so it is possible for MFVS to output a highly weighted vertex set, even higher than the weight of the calculated WMFVS (Gurobi may not always give a real optimal result because of its numerical instability). However, our WMFVS method clearly has better stability

The WFVS method returned an FVS with 528 vertices, which is approximately 14% larger than the size of the MFVS. The selected WFVS has a better average weight than both the MFVS and WMFVS. This result is consistent with our purpose for WFVSs, which focuses on the total weight rather than the size of the FVS.

Then, we used the five prepared cancer-related gene data sets to evaluate the results of these three methods. We verified the recall of the three FVS methods in the five data sets. The results are shown in Figure 3.7 and Table 3.3.

We can see that WMFVS and WFVS have better recall than traditional MFVS in all five sets, which is a benefit of the well-defined gene weights (especially for WFVS). Furthermore, we calculated the p-values of WMFVS and WFVS for 875 random MFVSs. See Table 3.4.

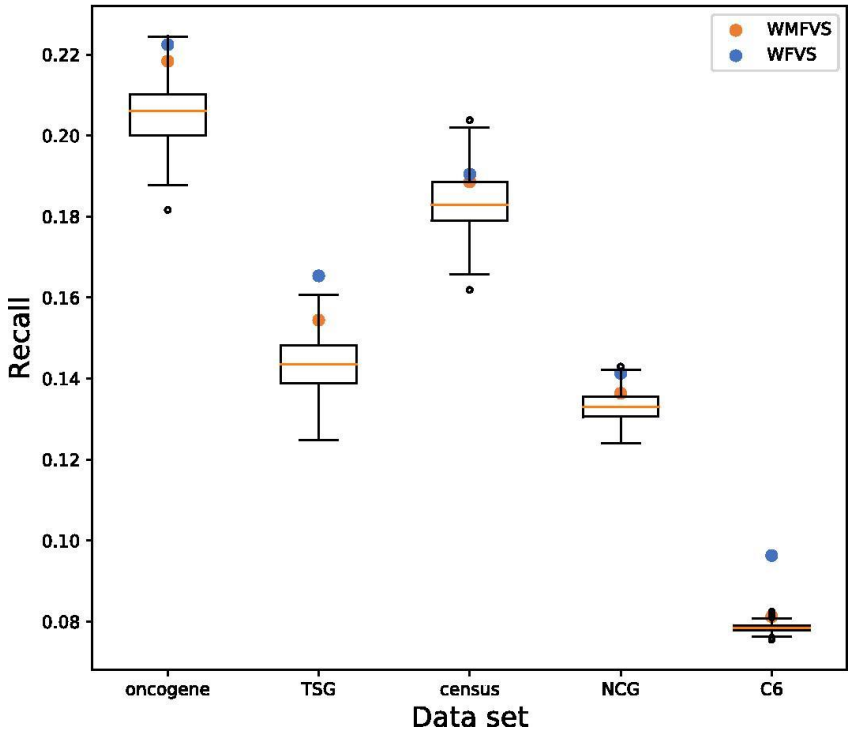


Figure 3.7 Distributions of the recalls for all the methods on different cancer gene data sets. The random MFVSs (boxplot), the WMFVS method (orange circle), and the WFVS method (cyan circle).

Table 3.3 The recall of each method in different gene sets.

	ONGene	TSGene	CGC	NCG	MSigDB
MFVS (average)	20.6%	14.4%	13.3%	13.2%	7.9%
WMFVS	21.8%	15.4%	18.8%	13.6%	8.1%
WFVS	22.2%	16.5%	19.0%	14.1%	9.6%

Table 3.4 The p-values of WMFVS and WFVS for random MFVSs.

	ONGene	TSGene	CGC	NCG	MSigDB
WMFVS	0.0491	0.0434	0.2537	0.2011	0.0069
WFVS	0.0069	0.0	0.1771	0.0091	0.0

For a certain data set, denote the recall of WMFVS by R_0. The recalls of all random MFVSs compose a set $\mathbb{R}$. Then the p-value of WMFVS is calculated by the following formula:

$$p_{WMFVS} = \frac{|\{R|\ R \geq R_0, R \in \mathbb{R}\}|}{|\mathbb{R}|}$$

The calculation of the p-value of WFVS is the same as above.

Next, as control methods, we considered several other kinds of methods of cancer gene prediction.

(1) Randomly select 463 genes (select 100 times and take the average performance).

(2) Select the 463 highest-weighted genes, which is a traditional differential expression-based method.

(3) Select the set of genes that appear in at least 49.5% MFVSs (we used 49.5% since the number of genes was exactly 463).

Table 3.5 The recalls (and precisions) of all the methods.

	463 random genes	Top 463 weighted genes	Genes appearing in 49.5% MFVSs	WMFVS (size: 463)	WFVS (size: 528)
ONGene	36(7.8%)	53(11.4%)	101(21.8%)	107(**23.1%**)	**109**(20.6%)
TSGene	46(9.9%)	66(14.3%)	93(20.1%)	99(**21.4**)	**106**(20.1%)
CGC	38(8.2%)	34(7.3)	99(**21.4%**)	99(**21.4%**)	**100**(18.9%)
NCG	88(19.0%)	102(22.0%)	162(35.0%)	165(**35.6%**)	**171**(32.4%)
MSigDB	308(66.5)	**405**(**87.5%**)	328(70.8%)	340(73.4%)	403(76.3%)

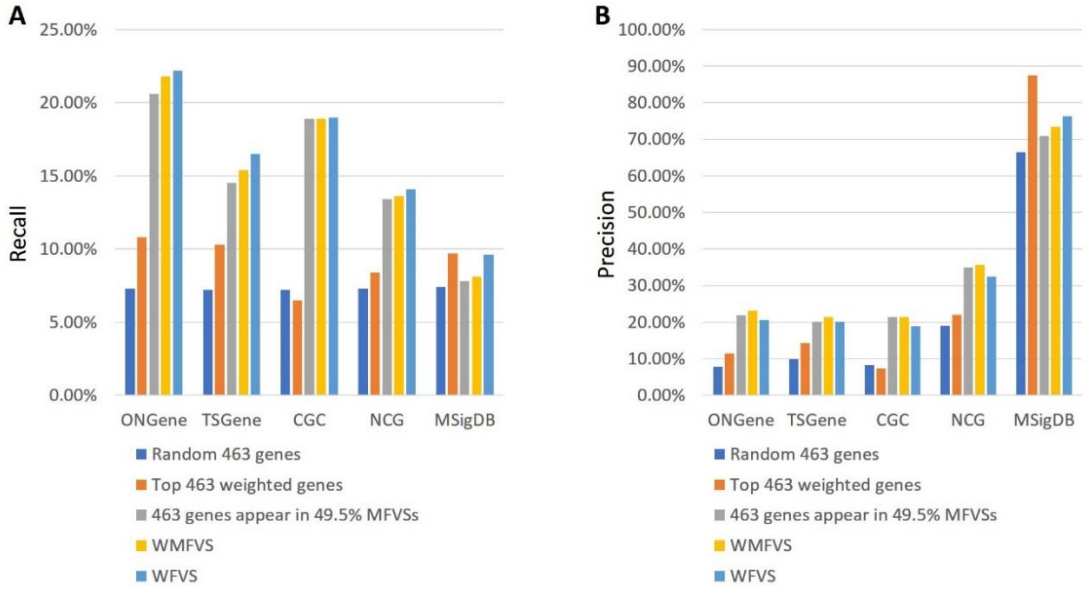


Figure 3.8 The recalls and precisions of all the methods.

Method (2) uses only weights for classification (i.e., a pure differential expression analysis method), while method (3) uses only graph theoretic results (i.e., a pure network analysis method). Method (3) selects the most common genes that appear in the MFVS. Intuitively, these genes should have great significance in the graph topology. The recalls and precisions of all these methods are listed in Table 3.5. Additionally, see Figure 3.8.

3.4. Discussion

3.4.1. Performance and enrichment score

In ONGene, TSGene and MSigDB, both WMFVS and WFVS have good p-values, but for CGC and NCG, the p-value is relatively high. One major reason is that there exists some correlation between the classification metric of the data set and the defined gene weight. To analyze this correlation, we utilized the enrichment score (ES) from GSEA [43], which reflects the degree to which a set S is overrepresented at the extremes (top or bottom) of an entire ranked list.

First, we sorted all the genes from the network by weight from high to low. Then, for a certain cancer gene set S, we traversed the sorted gene list, increasing a running-sum statistic when we encountered a gene in S and decreasing it when we encountered a gene not in S. We modified the increment and decrement value to ensure that the running sum was 0 at the end of the gene list. The enrichment scores of the five data sets are shown in Figure 3.9.

It is easy to see that the ONGene, TSGene and MSigDB data sets are significantly enriched at the tops of the lists. Although NCG seems enriched at the top, its ES is relatively low; the ES of CGC is even worse than that of NCG. The best enriched data set is MSigDB. Since this data set was constructed directly from microarray gene expression data from cancer gene perturbations, it is closely related to differential expression values. The ES value explains the different performances of WMFVS and WFVS in different data sets.

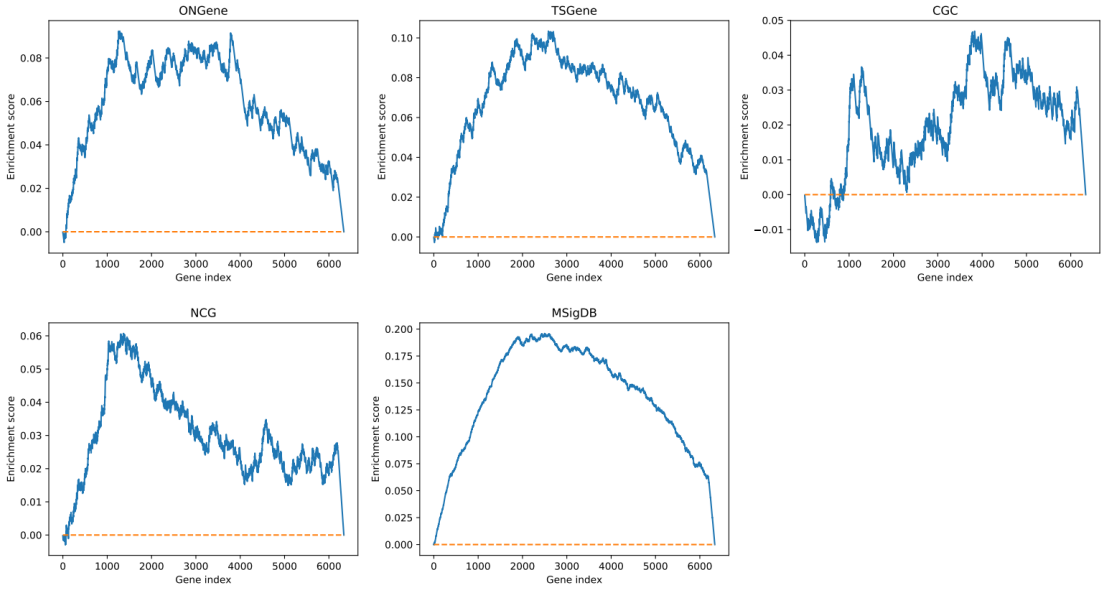


Figure 3.9 The enrichment score of each data set.

Table 3.5 and Figure 3.8 show that, except in MSigDB, WMFVS has the best precision and WFVS has the best recall. In MSigDB, cancer genes are closely related to the differential expression values of genes in breast cancer, leading to a precision of 87.5% for the simple weight-based method (i.e., method (2)). In this case, integration of the network structure may decrease the precision. However, in most cases, it is hard to find such a closely related metric for classification. We can observe that in other data sets, method (2) performs worse than the other methods. The results support the effectiveness of our WMFVS and WFVS methods.

3.4.2. Dark genes

As mentioned previously, traditional differential expression-based methods are not able to find graph-level important genes that have low differential expression values, i.e., dark genes. In our research, we defined a dark gene as a gene that has a relatively low weight (i.e., a low differential expression value) but is recorded as a cancer gene in the cancer gene data base(s). Specifically, we first derived the differentially expressed genes (DEGs)

by using the criteria of $|\log_2 FC| \geq 1$ and adjusted p-value≤ 0.05 from the TCGA breast cancer RNA-seq data, where FC is the fold change value of a certain gene. Based on these criteria, we found 4,245 DEGs (called the DEG set). Next, we curated the dark gene set from each cancer gene data set by excluding these DEGs.

In our experiments, we further selected the top 463 of the highest-weighted genes (i.e., the most differentially expressed genes; called the top-463 DEG set) to avoid an unbalanced gene number in comparison to the WMFVSs and WFVSs identified by the WMFVS and WFVS methods, respectively. For each of the cancer gene data sets, the precisions of the all-DEG set, top-463 DEG set, WMFVS and WFVS are shown in Figure 3.10.

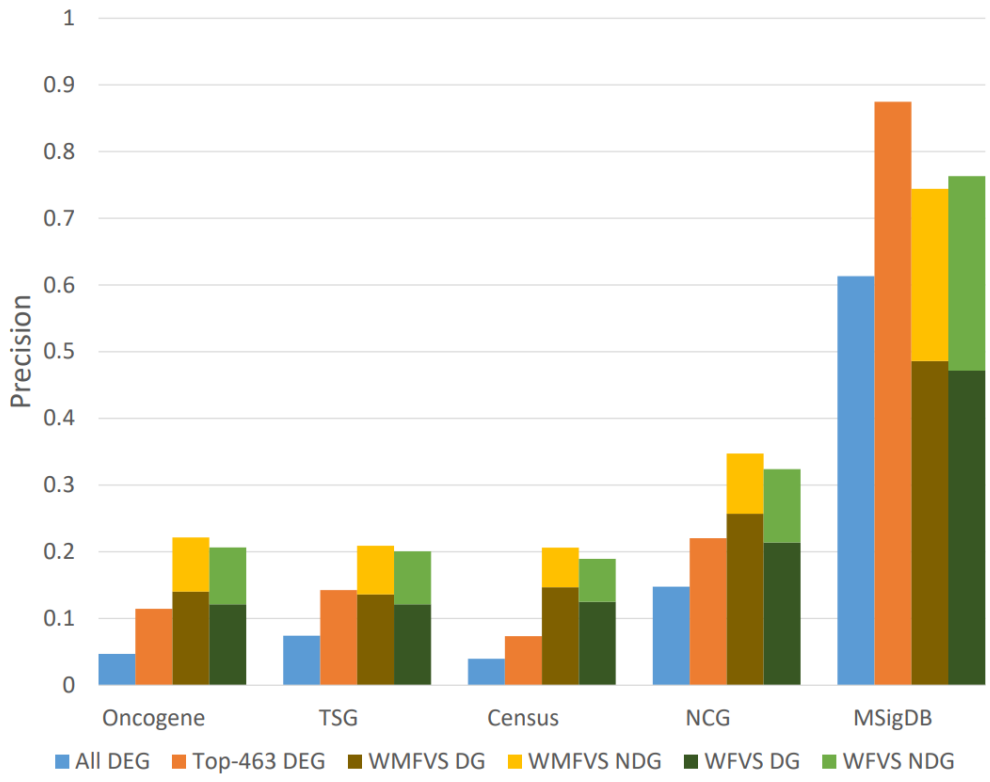


Figure 3.10 Comparison of the precision of the all-DEG set, top-463 DEG set, WMFVS set and WFVS set in five different cancer gene data sets. 'DG' and 'NDG' represent the ratios of dark genes and non-dark genes, respectively. Note that the all-DEG and the top-463 DEG sets contain no dark genes.

Figure 3.10 shows that our WMFVS and WFVS methods display better precision than the traditional DEG-based method (i.e., the all-DEG set and the top-463 DEG set) in four of five cancer gene data sets. Moreover, approximately 60%-70% of the genes are dark genes, which were detected by using our WMFVS and WFVS methods but ignored by the traditional DEG method. Even for the MSigDB C6 data set, which was generated directly from microarray data or from internal unpublished profiling experiments involving the perturbation of known cancer genes, the WMFVS and WFVS methods also have a good ability to detect dark genes. In summary, our WMFVS and WFVS methods have an advantage in identifying dark genes that are hard to find by using traditional DEG methods.

3.4.3. Missing-data cases

In this study, to retain the topological structure of the network, the weight-loss genes are assigned default weights of 0 rather than being removed. By further analysis, we found 3 weight-loss genes (i.e., CDC2, ZBTB8 and TADA3L) included in the WMFVS result, 7 weight-loss genes (i.e., CDC2, ZBTB8, RhoGDI, TADA3L, RNF12, NP and MAP3K7IP1) contained in at least one of the 875 random MFVS results, and no weight-loss genes in the WFVS result. In particular, CDC2 and ZBTB8 were included in all the random MFVS results as well as in the WMFVS result. The CDC2 gene is related to the highly conserved protein CDK1, which functions as a serine/threonine kinase and is a key player in cell cycle regulation [46]. The CDC2 gene is also considered a cancer-related gene whose overexpression may play an important role in human breast carcinogenesis [47]. While little is known about the ZBTB8 gene, the same ZBTB family protein, ZBTB7A, has been implicated in high expression in cancer tissue and the breast cancer cell lines MDA-MB-231 and MCF-7 [48], suggesting that ZBTB8 may act as a transcriptional repressor or be involved in tumorigenesis. The uncovering of CDC2 and ZBTB8 genes illustrates that the WMFVS method may address the disadvantage of traditional DEG methods in missing-data cases.

3.5. Summary

We present several new methods for cancer gene prediction. Our WMFVS method uses differential gene expression to select MFVSs, improving the stability of the general MFVS algorithm and obtaining a much better result than the differential gene expression-based method when the weights of the genes are well defined. Our WFVS method is a variant of WMFVS, which aims at finding an FVS in the network that contains the maximum total weight. This method obtains better recall than WMFVS by sacrificing precision. Thus, generally, if the researcher wants to reveal as many potential cancer genes as possible, WFVS is better; if the researcher prefers better precision, then WMFVS is better. Furthermore, since WFVS ignores the restriction of the output size, it focuses more on the vertex weight than WMFVS. Therefore, if the researcher has good confidence in the weight definition, i.e., the weights are closely related to the classification, WFVS will have a better result than WMFVS. We can see this from the data analyses on the MsigDB data set, which has the highest enrichment score on our defined weights. However, in many cases, since we are not sure whether the defined weights are closely related to the classification, using WMFVS will maintain better precision for the prediction.

Chapter 4

Densest subgraph-based methods for protein-protein interaction hot spot prediction

4. Densest subgraph-based methods for protein-protein interaction hot spot prediction

4.1. Background

Proteins realize their functions by interacting with other proteins and/or chemical compounds [49]. Protein-protein interactions play crucial roles in most biological processes. In a protein-protein binding interface, the binding free energy is not uniformly distributed among the residues. Instead, there are hot spots, which contribute most to the binding energy in protein interfaces [50]. Detecting hot spots in protein-protein interactions is meaningful in regulating protein-protein binding and may also contribute to disease control and drug design. Experimentally, a hot spot residue is defined as having a change in binding energy $\Delta\Delta G \geq 2.0$ kcal/mol upon its mutation to alanine [51]. Several databases have been constructed to collect experimental hot spots from alanine scanning mutagenesis experiments, and two famous databases are the Alanine Scanning Energetics Database (ASEdb) [13] and the Binding Interface Database (BID) [52]. Another widely used database is the SKEMPI database [53], which is new and continually updated (public access to ASEdb and BID is no longer supported). However, finding hot spots by experimental methods is time-consuming and costly; thus, a need for computational methods arises [54].

Several kinds of methods have been designed to predict hot spots. The first type is based on molecular dynamics simulations [55, 56]. Although these methods provide detailed analyses of protein interfaces and have good prediction results, they have difficulty dealing deal with large-scale data because of the high computational cost. Another kind of method is based on energy estimation [57, 58], which estimates the energetic contribution to binding for every interface residue to predict hot spots. Compared to molecular dynamics simulation, energy estimation methods are more

efficient in predicting hot spots from large protein complexes.

In recent years, machine learning methods have been frequently used in hot spot prediction, such as extreme gradient boosting [59], random forests [60], and support vector machines (SVMs) [61]. The advantage of machine learning based methods is that they can filter and utilize various possible features to classify residues, together with a well-designed model, and usually have high performance in hot spot prediction. However, since experimentally approved hot spot data are scarce, a large percentage of real hot spot residues are not recognized in hot spot datasets. In machine learning methods, the low rate of positive instances makes it difficult to train models. Additionally, in some methods such as [51, 62-64], to balance the ratio of positive instances to negative ones, only residues with less than 0.4 kcal/mol binding free energy are defined as non-hot spots, which further reduces the size of the training set.

On the other hand, there are some methods based on graph theory and network analysis. Tuncbag et al. transformed residue interaction networks into minimum-cut trees and then identified the high-degree nodes as hot spots [65]. Li et al. searched for bicliques from the input network to find highly connected patterns, which have a high possibility of forming a group of hot spots [66]. The graph theory-based methods do not need existing hot spot data to train the models, avoiding the need for many experimental resources, and the prediction results can be a good guide for further biological experiments. Unfortunately, the existing graph theory-based methods have very low recall. Although some hot spots can be precisely detected by these methods, many possible hot spots are ignored.

Here, we consider using other graph theory methods, which are based on the densities of subgraphs, to analyze residue interaction networks. Generally, high density refers to a high connectivity between vertices, and it often relates to binding sites in complexes. By further evaluation, we find that our methods have an obvious advantage in finding potential hot spots, as well as having similar precision to that of the existing methods. The results of these densest subgraph-based methods (DS-based methods) can be a good

reference for future bio-experiments.

Generally, a certain input network may contain multiple densest subgraphs. We can simply select one arbitrary densest subgraph as an output. In this research, we use DS to represent this method. However, because of this uncertainty, it is difficult to ensure the performance of the DS. To obtain better performance in practice, we propose three variant methods based on the DS method (DS-based methods). The first method yields all the minimal densest subgraphs [67] as the result, and we use Min-DS to denote this method. Compared to DS, Min-DS has no uncertainty and has better precision and recall than DS. The second method, Max-DS, is based on another concept, namely, the maximal densest subgraph [68]. The results of Max-DS include those of Min-DS, and it has higher recall but lower precision than Min-DS. To maximize the ability to find potential hot spots, we develop a third method, Min-SDS, which is also the main method in our research. This method is similar to Min-DS but has a weakened restriction in detecting the minimal densest subgraph. By further evaluation, we find that Min-SDS has the best recall and F2-score among all the graph theory-based methods and performs well in detecting unknown hot spots.

4.2. Methods

4.2.1. Problem transformation

For a given protein complex, we first convert the residue spatial coordinate information to an undirected graph, where the vertices correspond to the residues and the edges correspond to the contacts between residues. See Figure 4.1. The PDB id of the example in Figure 4.1 is 1JTG [69].

Then, the hot spot prediction problem is transformed into the problem of searching for critical vertices in an input graph, and the selected vertices correspond to the predicted hot spot residues.

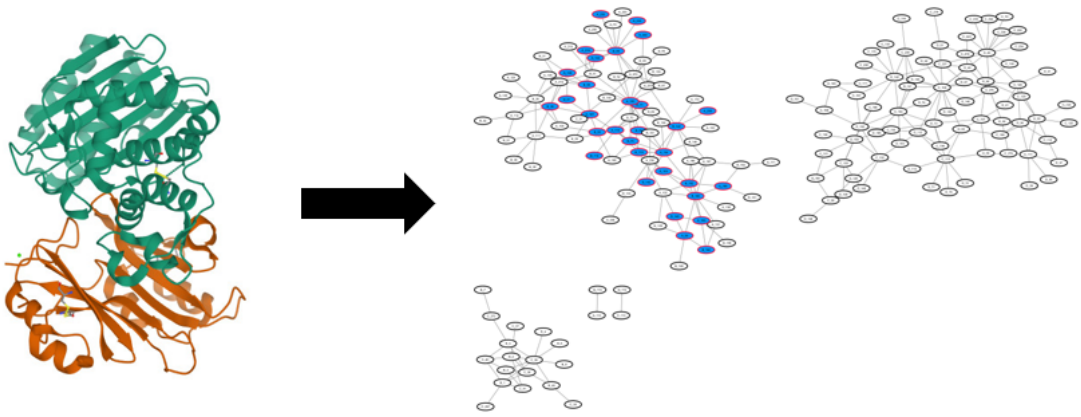


Figure 4.1 Transform the protein complex coordinate data to a residue interaction network. Only the residues that are close to some other residues in the other protein chain are kept in the network. An edge is added between two close residues from different chains.

4.2.2. Densest subgraph

Given an undirected graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ is the set of vertices and E is the set of edges of G. Let $S = (V', E')$ be a subgraph of G, where $V' \subseteq V$ and E' is the set of edges induced by V', then the density of S is defined by $\rho(S) = \frac{|E'|}{|V'|}$. If S has the maximum density among all possible subgraphs of G, then S is a densest subgraph of G, and this maximum density is defined as the density of the graph G, denoted by $D(G)$. A certain graph G may have multiple densest subgraphs. See more descriptions and examples of densest subgraph in Section 2.3.

In [70], a linear programming (LP)-based method was proposed to search for a densest subgraph of G. For each edge $(i, j) \in E$, a real-valued variable $0 \leq x_{i,j} \leq 1$ is set, and for each vertex $i \in V$, a real-valued variable $0 \leq y_i \leq 1$ is set. Then, the following LP method BasicLP returns a solution that contains the information of an arbitrary densest subgraph of G.

For an optimal solution of BasicLP, the set of vertices $S = \{i \in V \mid y_i > 0\}$ induces a densest subgraph of G. We also use DS to denote this LP-based method.

BasicLP(V, E)

Maximize $$\sum_{(i,j)\in E} x_{i,j}$$

Subject to $$x_{i,j} \le y_i \qquad \forall (i,j) \in E$$

$$x_{i,j} \le y_j \qquad \forall (i,j) \in E$$

$$\sum_{i\in V} y_i \le 1$$

$$x_{i,j} \ge 0, y_i \ge 0 \qquad \forall i,j$$

Furthermore, we have the following proposition:

Proposition 3. *For any optimal solution of BasicLP, the set of vertices* $S = \left\{ i \in V \middle| y_i \ge \frac{1}{|V|} \right\}$ *induces a densest subgraph of G.*

Proof. For any optimal solution, obviously, we have $\sum_{i\in V} y_i = 1$, thus if there exists some $y_i < \frac{1}{|V|}$, there must exist some $y_j > \frac{1}{|V|}$.

According to the proof of Lemma 4.1 of [67], when an optimal solution has different non-zero values of y_i's, if we remove the y_i's with the lowest non-zero value (set them to 0), the remaining y_i's with non-zero value(s) still correspond to a densest subgraph. We can repeat the process until the lowest non-zero value is larger than $\frac{1}{|V|}$, and the remaining y_i's with non-zero values correspond to a densest subgraph.

■

Accordingly, in practice, we select the vertices with $y_i \ge \frac{1}{|V|}$ rather than $y_i > 0$ as the output because of the numerical error of the Gurobi solver [12].

4.2.3. Minimal densest subgraph

Given an undirected graph $G = (V, E)$, let S be a densest subgraph of G. If for any subgraph S' of S, $\rho(S') < \rho(S)$, then S is a minimal densest subgraph. One graph may include multiple minimal densest subgraphs.

In [67], Balalau et al. presented an LP-based method to find all minimal densest subgraphs for an input graph. The method to find an arbitrary minimal densest subgraph can be divided to three parts as follows:

Algorithm $TryRemove(G, u, \rho_{max})$

Input: A graph $G = (V, E)$, a node u to be removed, the maximum density $\rho_{\max}$.
Output: Returns a densest subgraph in G not containing u, or $null$ if every densest subgraph must contain u.

$H \coloneqq$ BasicLP$(V - \{u\}, E)$
if $\rho(H) \geq \rho_{max}$**:**
    return H
else:
    return $null$

The algorithm $TryRemove(G, u, \rho_{max})$ tries to remove u from G, and then find a subgraph that has at least ρ_{max} density; if there does not exist such a subgraph, then return $null$.

Algorithm $TryEnhance(G, u, \rho_{max})$

Input: A graph $G = (V, E)$, a node $u \in V$, the maximum density $\rho_{\max}$.
Output: Returns a densest subgraph in G containing u with minimum cardinality, or $null$ if there is no densest subgraph containing u.

Add constraints $\sum x_{ij} = \rho_{max}$ and $y_u \geq \frac{1}{n}$ to the *BasicLP*, then change the objective function to **Maximize** y_u, call the modified LP as *BasicLP'*. The return of $BasicLP'$ is a feasible subgraph, or $null$ if no feasible result exists.
return $BasicLP'(G)$

The BasicLP' is as follows:

BasicLP'(G):

Maximize y_u

Subject to $x_{i,j} \leq y_i \quad \forall (i,j) \in E$

$x_{i,j} \leq y_j \quad \forall (i,j) \in E$

$$\sum_{i \in V} y_i \leq 1$$

$x_{i,j} \geq 0, y_i \geq 0 \quad \forall i,j$

$\sum x_{ij} = \rho_{max}$

$y_u \geq \frac{1}{n}$

The algorithm $TryEnhance(G, u, \rho_{max})$ tries to find a subgraph with density at least ρ_{max} and must contain u as one of the vertices in the subgraph; if no such a subgraph exists, then return $null$. Note that, since the objective function is to maximize the value of y_u, if a feasible solution exists, the result of *TryEnhance* will always be a minimal densest subgraph.

Based on $TryRemove$ and $TryEnhance$, we finally have the following algorithm:

Algorithm *FindMinimal*(G)

Input: A graph $G = (V, E)$.
Output: A minimal densest subgraph in G.

$H(\bar{V}, \bar{E}) = BasicLP(G)$, let ρ_{max} be the density.

while True:

  select $u \in \bar{V}$ **arbitrarily**

  $H_1(V_1, E_1) = TryRemove(H, u, \rho_{max})$

  $H_2(V_2, E_2) = TryEnhance(H, u, \rho_{max})$ # Note that H_2 will never be $null$.

  if $H_1 == null$ **then return** H_2

  if $|V_1| < |V_2|$ **then** $H = H_1$ **else** $H = H_2$

return H

Figure 4.2 shows an explanation of the algorithm $FindMinimal$.

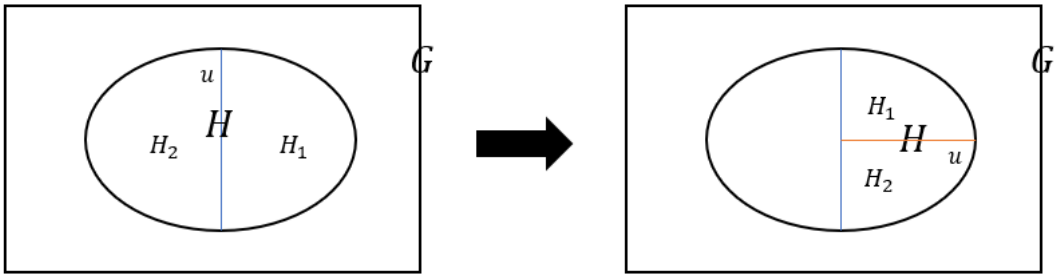


Figure 4.2 An explanation of the algorithm $\boldsymbol{FindMinimal}$. In the left figure, assume $\boldsymbol{H_1}$ is not $\boldsymbol{null}$, and its size is smaller than $\boldsymbol{H_2}$, then $\boldsymbol{H}$ is assigned to $\boldsymbol{H_1}$, which decreases the problem scale to the right figure. Repeat the process until $\boldsymbol{H_1} == \boldsymbol{null}$.

Once we find a minimal densest subgraph from G by algorithm *FindMinimal*, we record it and remove it from G. Then repeat the process until no densest subgraph can be found. The whole method is as follows:

1. $result := \emptyset$.
2. Find a minimal densest subgraph R by *FindMinimal*.
3. If $\rho(R) < D(G)$, then return result; otherwise, set $result = result \cup R$, remove R from the graph, and then jump to step 2.

This method can find all the minimal densest subgraphs, and we use Min-DS to denote this method.

4.2.4. Maximal densest subgraph

Given an undirected graph $G = (V, E)$, let S be a densest subgraph of G. If any densest subgraph of G is a subgraph of S, then S is the maximal densest subgraph.

Proposition 4. *For any undirected graph, exactly one maximal densest subgraph exists.*

Proof. For a given graph G, the union of all possible densest subgraphs is obviously a maximal densest subgraph. However, there may exist another maximal densest subgraph.

Assume more than one maximal densest subgraph exists. Let S_1 and S_2 be two different maximal densest subgraphs. According to Corollary 4.1 of [67], $S_3 := S_1 \cup S_2$ is also a densest subgraph. Since $S_1 \neq S_2$, S_3 is not a subgraph of S_1 or S_2, thus S_1 and

S_2 are not maximal densest subgraphs, which contradicts to the assumption. Therefore, only one maximal densest subgraph exists.

■

In [68], a binary search-based method is introduced to find the maximal densest subgraph. Here we propose two novel linear programming methods to solve this problem.

We can find the maximal densest subgraph of an input graph $G = (V, E)$ by an integer linear programming (ILP)-based method. For each edge $(i, j) \in E$, we set a real-valued variable $0 \leq x_{i,j} \leq 1$; for each vertex $i \in V$, we set a real-valued variable $0 \leq y_i \leq 1$ and a Boolean variable z_i. Let D be the maximum density of G (we can obtain D by BasicLP). Then, we have the following ILP:

MaxILP

Maximize $\sum_{i \in V} z_i$

Subject to

$$x_{i,j} \leq y_i \qquad \forall (i,j) \in E \qquad (1)$$

$$x_{i,j} \leq y_j \qquad \forall (i,j) \in E \qquad (2)$$

$$\sum_{i \in V} y_i \leq 1 \qquad (3)$$

$$x_{i,j} \geq 0, y_i \geq 0 \qquad \forall i, j \qquad (4)$$

$$\sum_{(i,j) \in E} x_{i,j} \geq D \qquad (5)$$

$$y_i - \frac{z_i}{|V|} \geq 0 \qquad \forall i \in V \qquad (6)$$

This ILP method is denoted as Max-DS. Furthermore, we have Proposition 5.

Proposition 5. *For an optimal solution $H = (x^H, y^H, z^H)$ of MaxILP, the set of vertices $\{i | z_i \in z^H, z_i = 1\}$ induces the maximal densest subgraph of G.*

Proof. Let S be the maximal densest subgraph, then the following solution (x^S, y^S, z^S) is a feasible solution of MaxILP:

$$x_{i,j}^S = \begin{cases} \frac{1}{|S|} & \text{if both } i \in S \text{ and } j \in S \\ 0 & \text{otherwise} \end{cases}$$

$$y_i^S = \begin{cases} \frac{1}{|S|} & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases}$$

$$z_i^S = \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases}$$

Constraints (1)-(5) restrict the subgraph induced by $\{i | i \in V, y_i \geq \frac{1}{|V|}\}$ be a densest subgraph (by Proposition 3). By constraint (6), $z_i = 1$ only if $y_i \geq \frac{1}{|V|}$. Thus $\sum_{i \in V} z_i \leq |S|$, otherwise, the set $\{i | i \in V, y_i \geq \frac{1}{|V|}\}$ induces a densest subgraph that has a size larger than $|S|$, which is impossible.

■

We can also use an LP-based method to find the maximal densest subgraph. First, we modify BasicLP to the following LP (the definition of the variables is the same as in BasicLP):

MaxLP(V, E, D, R)

Maximize $\sum_{(i,j) \in E} x_{i,j}$

Subject to

$$x_{i,j} \leq y_i \qquad \forall (i,j) \in E \qquad (7)$$

$$x_{i,j} \leq y_j \qquad \forall (i,j) \in E \qquad (8)$$

$$\sum_{i \in V} y_i \leq 1 \qquad (9)$$

$$x_{i,j} \geq 0, y_i \geq 0 \qquad \forall i, j \qquad (10)$$

$$\sum_{(i,j) \in E} x_{i,j} \geq D \qquad (11)$$

$$y_i \geq \frac{1}{|V|} \qquad \forall i \in R \qquad (12)$$

$$\sum_{i \in V - R} y_i \geq \frac{1}{|V|} \qquad (13)$$

Here, D is the density of the input graph, and R is a subset of V. Compared to BasicLP, we add constraints (11)-(13) to the program. Constraint (11) requires that the solution leads to a densest subgraph; constraint (12) requires that all the vertices in R should be selected to the solution; constraint (13) requires that at least one vertex that is not in R should be selected. We set the objective value as the return of BasicLP and use $\{i|i \in V, z_i = 1\}$ or $\emptyset$ (if no feasible solution is found) as the return of MaxLP.

Then, we have the algorithm *FindMaximal*.

Algorithm *FindMaximal*(V, E)

```
D := BasicLP(V, E)
R := ∅
while True do:
    R' = MaxLP(V, E, D, R)
    if R' == ∅ then:
        return R
    else:
        R = R ∪ R'
```

The correctness of *FindMaximal* is obvious. In the worst case, we need to run MaxLP $O(n)$ times, and thus we can solve the problem in polynomial time.

In practice, the MaxILP and *FindMaximal* methods have very similar time costs, and thus the evaluation is based on the results of MaxILP, which is easier to implement (although both methods have the same results because of the uniqueness of the maximal densest subgraph).

4.2.5. Minimal sub-densest subgraph

In some protein complexes, multiple binding interfaces may exist, while in the residue interaction network, the interface areas may have different densities. If we always search the densest subgraph, some hot spots in some binding interfaces may be ignored.

Here, we consider weakening the restrictions of in Min-DS to find more potential hot spots. The skeleton of Min-DS is as follows [67]:

1. $result := \emptyset$.
2. Find a minimal densest subgraph R.
3. If $\rho(R) < D(G)$, then return result; otherwise, set $result = result \cup R$, remove R from the graph, and then jump to step 2.

In step 3, if Min-DS has a smaller density than the input graph, then the process stops. Here, we consider adding a tolerance θ to step 3 as follows:

1. $result := \emptyset$.
2. Find a minimal densest subgraph R.
3. If $\rho(R) < \theta * D(G)$, where $0 < \theta < 1$, then return result; otherwise, set $result = result \cup R$, remove R from the graph, and then jump to step 2.

We call the result the minimal sub-densest subgraphs, and this method is named Min-SDS.

4.2.6. Biclique

In [66], a biclique-based method is proposed to predict hot spots. Given a bipartite graph $G = (V_1, V_2, E)$, where V_1 and V_2 are two distinct vertex sets, and E is the set of edges in which only edges between V_1 and V_2 exist. A biclique $B = (V_1', V_2', E')$ is a subgraph of G, that $\forall v_i \in V_1'$ and $\forall v_j \in V_1'$: $(v_i, v_j) \in E'$. If a biclique is not included by any other biclique, then this biclique is called a maximal biclique. Figure 4.3 shows an example.

Furthermore, we have the following observations:

- A bipartite graph may have multiple maximal biclique subgraphs;
- Some vertices may be contained in different biclique subgraphs;
- The size of maximal biclique subgraphs can be different.

Using the LCM-MBC algorithm, we can find all maximal biclique subgraphs of a given bipartite graph in linear time.

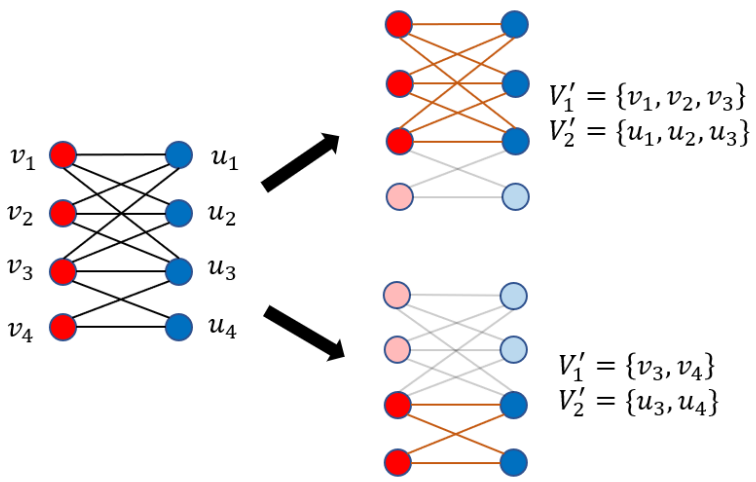


Figure 4.3 Two bicliques (right part, deep color subgraphs) of an input graph. Both of the bicliques are maximal.

In [66], researchers first construct the residue interaction networks for the protein complexes, then they find all the bicliques with at least 3 vertices in each side. The found biclique patterns are then searched in all complexes, and only the patterns that appear in at least 5 complexes are recognized as hot spots.

In our research, since we only use one network to predict hot spot, we skipped the pattern searching phase of the biclique method in practice, which should increase the recall but decrease the precision in evaluation.

4.2.7. Minimum cut tree

In [65], researchers transform the residue interaction network to minimum cut trees (mincut trees), then the high-degree tree nodes are recognized as hot spots.

Given an undirected, connected graph $G = (V, E)$, a cut of G is a partition of the node set into two sets, and consists of all edges that have one endpoint in each partition. Let $s, t \in V$, an s-t cut is defined as a cut, which puts s and t into different node sets of the partition. Figure 4.4 shows some examples of s-t cuts.

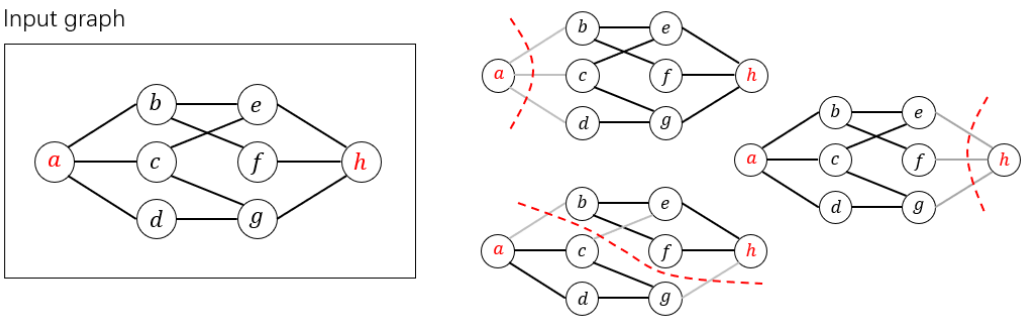


Figure 4.4 Three possible cuts of a and h in the input graph.

A minimum weight s-t cut (min *s*-*t* cut, or mincut of *s*-*t*) is defined as an s-t cut that has the minimum total weight. Based on the max-flow min-cut theory [71], i.e. The weight of min s-t cut always equals to the max-flow from s to t, we can use the Ford-Fulkerson algorithm [71], which finds the maximum flow between two vertices, to efficiently find the mincut of any vertex pair.

Gomory and Hu introduced a tree structure (Gomory-Hu tree, or mincut tree) [72] that shows all the mincut between any pair of vertices in a graph. This tree can be computed using only $n-1$ min cut computations, where n is the number of vertices.

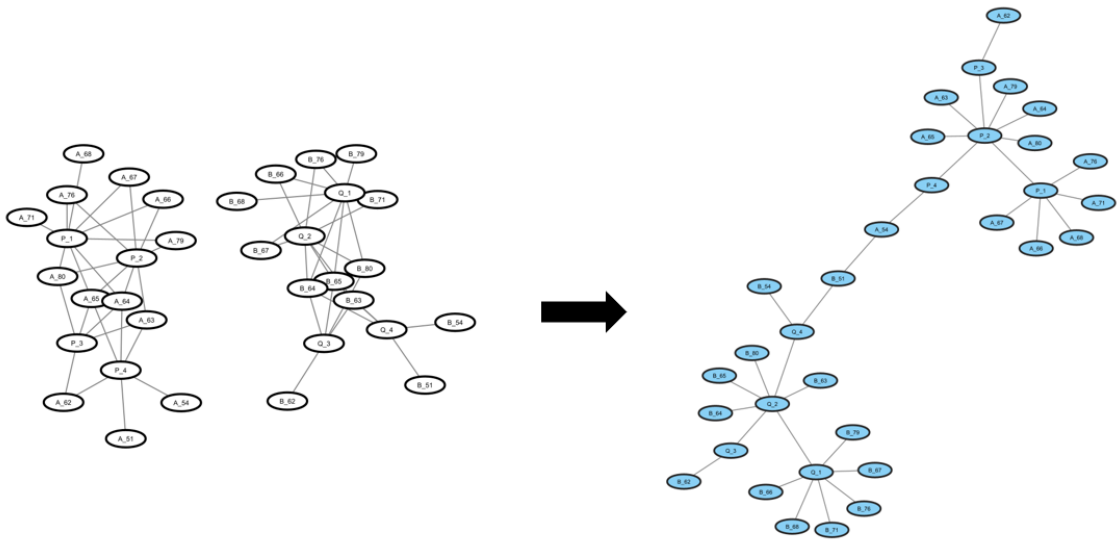


Figure 4.5 Residue interaction network (left) and its mincut tree (right) of the complex 3UIH.

In [65], the edges in the residue interaction network are weighted by solvent-mediated potential data, while in our research edges are unweighted. However, we can still apply the mincut methods by giving all edges a weight of 1. The following graph shows a result of compressing a graph to a mincut tree.

In the mincut tree in Figure 4.5, we can obviously find some nodes that have higher degrees than the other nodes. In [65], the tree nodes with a degree higher or equal to 3 are predicted as hot spots.

4.3. Result

4.3.1. Dataset

We mainly use the data from the SKEMPI 2.0 dataset [53], which records 7085 pieces of mutation information on 341 protein complexes, to define the hot spots in protein complexes. Specifically, if a residue has $\Delta\Delta G = \Delta G_{mut} - \Delta G_{wt} \geq 2.0$ kcal/mol in an alanine-mutation experiment, then this residue is recognized as a hot spot [13]. Here, ΔG_{wt} and ΔG_{mut} are the binding free energies upon complex formation of the wild-type and alanine-mutated proteins, respectively. ΔG can be calculated by $\Delta G = RT \ln Kd$, where R is the ideal gas constant, T is the absolute temperature, and Kd is the affinity of the wild-type (wt) or mutant (mut) complexes. Thus, we have [73]:

$$\Delta G_{wt} = \left(\frac{8.314}{4184}\right) * (273.15 + 25.0) * \ln(wt)$$

$$\Delta G_{mut} = \left(\frac{8.314}{4184}\right) * (273.15 + 25.0) * \ln(mut)$$

The residue interaction network data are based on PDB spatial data [74]. In a protein complex, any two residues in different chains are regarded as contacting each other if there exist two atoms a and b from each residue such that their distance $d(a,b) \leq r_a + r_b + 2.75$Å, where r is the van der Waals radius, and 2.75Å is the diameter of a water molecule [66]. See Figure 4.6.

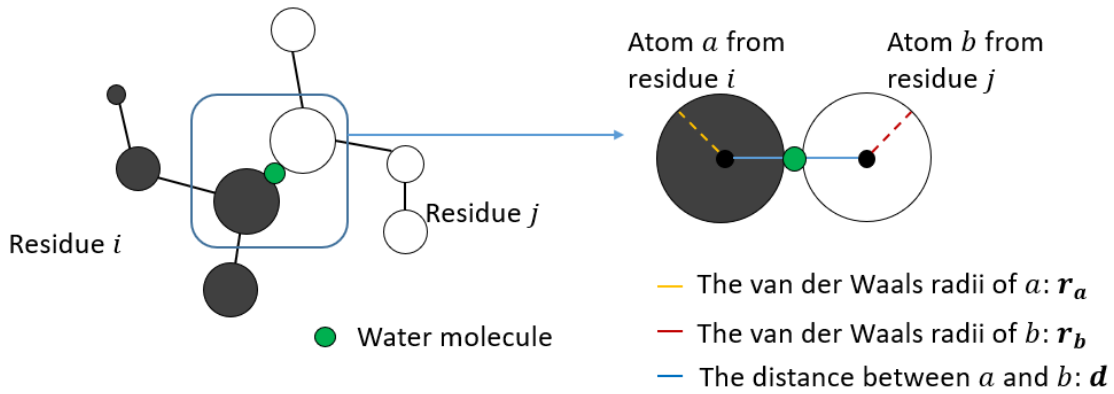


Figure 4.6 Two residues are determined in contact if they are close enough to each other. Specifically, if we can find a pair of atoms a and b from residues i and j, respectively, that $d < r_a + r_b + 2.75$, then residue i and j are in contact.

To build a residue interaction network for each protein complex, only the residues that contact at least one other residue are selected as vertices of the network, and an edge is added between any two contacting vertices.

The atom spatial data in PDB are based on crystal artifacts, sometimes they may not directly reflect the natural protein quaternary structure of complexes [75]. See Figure 4.7.

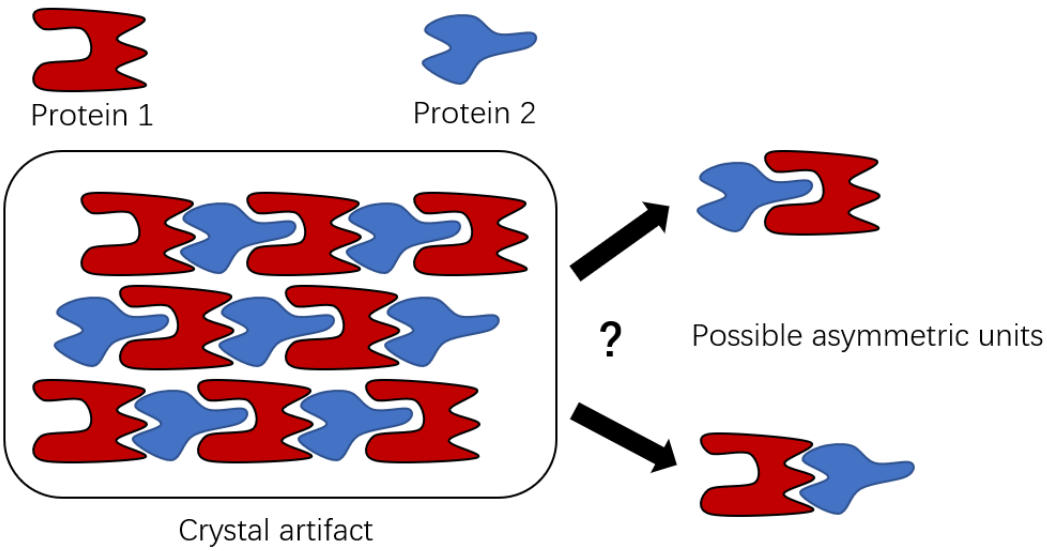


Figure 4.7 One crystal artifact sample may include several possible asymmetric units.

To avoid the problem of choosing proper biological assemblies among asymmetric

units, we selected 223 complexes from the 341 complexes, each of which has only one possible biological assembly, to construct residue interaction networks. We further selected 67 networks with at least 3 bio-experimentally approved hot spots for evaluation.

In addition, we used another independent hot spot dataset, AB-bind [76], for evaluation, which contains 1101 mutation records on 27 complexes. Using the same data selection strategy, 5 complexes were selected for result evaluation.

4.3.2. Experiments and evaluation

We implemented the DS, Min-DS, Max-DS, Min-SDS, Biclique, and Mincut methods on the built networks.

The DS method finds an arbitrary densest subgraph of the input network; the Min-DS method finds all the minimal densest subgraphs [67]; the Max-DS method finds the maximal densest subgraph; and the Min-SDS method finds a set of nonintersecting subgraphs with high densities.

Biclique and Mincut are existing methods. The Biclique method [66] finds all the bicliques of the input network. In our experiments, only the bicliques that contain at least 3 vertices on each side are selected as the result. The Mincut method [65] first builds the mincut tree of the input network, and then the high-degree (at least degree 3) nodes in the tree are selected as the result.

Let TP, TN, FP and FN be the numbers of true positive, true negative, false positive and false negative residues in the predictions, respectively. The standard metrics are defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{F2-Score} = \frac{5 \times \text{Precision} \times \text{Recall}}{4 \times \text{Precision} + \text{Recall}}$$

The average results of all the six methods are shown in Figure 4.8 ($\theta = 0.85$ for Min-

SDS).

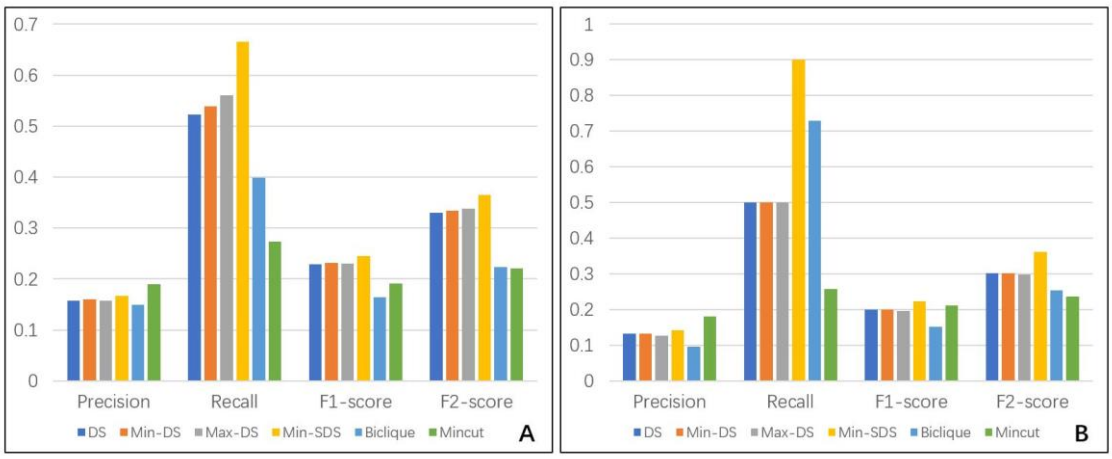


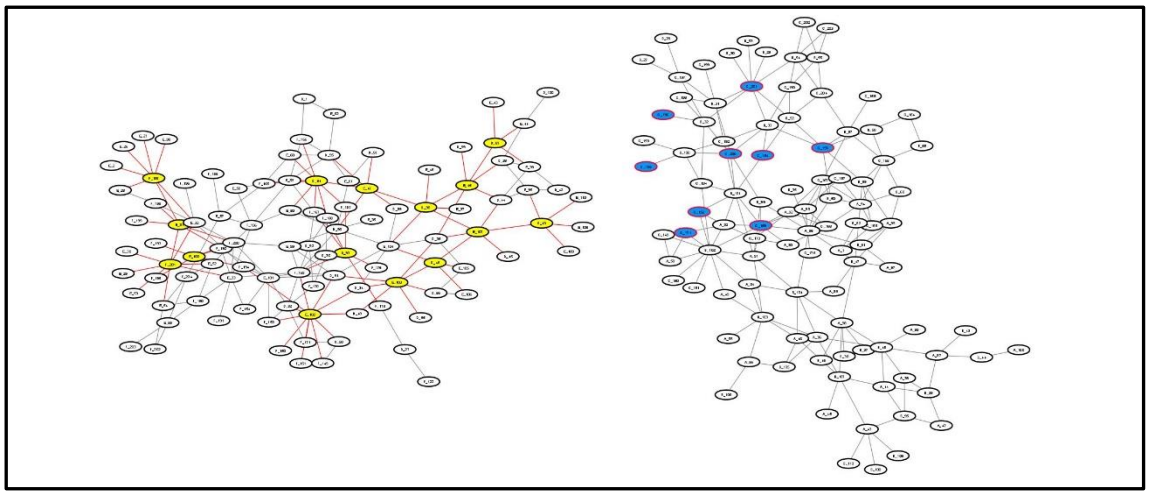
Figure 4.8 Clustered column chart of the performances of each method on SKEMPI (A) and AB-bind (B). The result distributions on the two charts are similar. In both datasets, Min-SDS has the best recall and F-score, and all DS-based methods outperform the existing methods in terms of F2-score.

Compared to the existing methods, our DS-based methods have much better F-scores. Although Mincut has the best precision, its recall is very low compared to the other methods. In hot spot research, there is a lack of bio-experiments detecting whether a residue is a hot spot. Even if some experiments on a residue have been performed and indicated $\Delta\Delta G < 2.0$ kcal/mol, it is difficult to determine that this residue is not a hot spot. Many potential hot spots may be false-negatively tagged by bio-experiments. In this situation, higher recall should be more beneficial than higher precision.

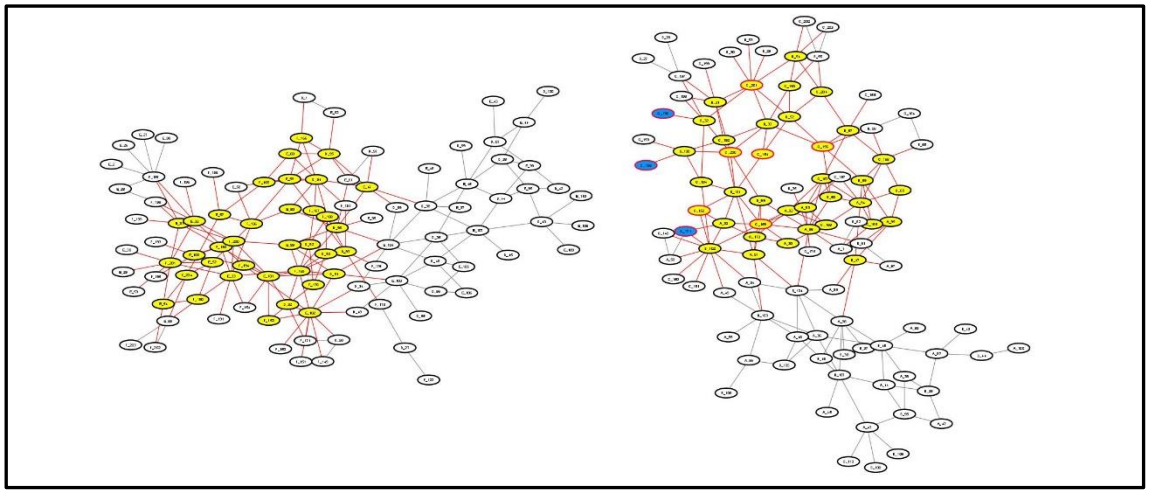
Another disadvantage of Mincut is that its results tend to be in one connected component. However, a protein complex may have multiple binding sites, which means that several distinct subgraphs may contain hot spots, while in most cases, the Mincut method focuses on only one of them.

As an example, complex 1AHW [77] consists of 3 molecules, and each molecule has 2 chains (AD, BE and CF). These 6 chains compose a heterohexamer (preferred) biological assembly composition. By checking the residue interaction network, 5 large connected subgraphs are found to exist: A-B, D-E, A-F, A-B-C and D-E-F (subgraph A-

B means that all the residues in this subgraph come from chain A or B, and the other terms have similar meanings). In these subgraphs, A-B-C and D-E-F are highly connected, and both of them have high possibilities of containing hot spots. In fact, all the experimentally approved hot spots are gathered in the A-B-C subgraph. However, in practice, the Mincut method only predicts residues in the D-E-F area and thus performs poorly in this instance. For details, see Figure 4.5.



(a) Mincut



(b) Min-SDS

Figure 4.9 The results of Mincut and Min-SDS on the graph of complex 1AHW. TP: red outline, yellow fill; FP: black outline, yellow fill; TN: black outline, white-fill; FN: red outline, blue fill. These figures show only part of the 1AHW network.

Our DS-based methods, especially the Min-SDS method, are not restricted to only one connected area, and thus all highly connected areas can be selected. For the instance 1AHW, the result of the Min-SDS method distributes in both A-B-C and D-E-F subgraphs, successfully covers the approved hot spots, and predicts the possible hot spots in the D-E-F area.

Since the Min-SDS method removes the restriction of 'densest', it has the best advantage in finding possible hot spots. In our experiments, the tolerance θ of Min-SDS was set to 0.85; i.e., all minimal subgraphs with a density higher than $0.85 * D$ were selected, where D is the maximum density of the input graph.

We tested the performance of Min-SDS on different θ values from 0.5 to 1.0, and the results are shown in Figure 4.10. With the decrease in θ, the precision decreases while the recall increases. The F2-score peaks when $\theta = 0.85$; this score is the best among those of all DS-based methods, and is obviously better than those of the existing methods.

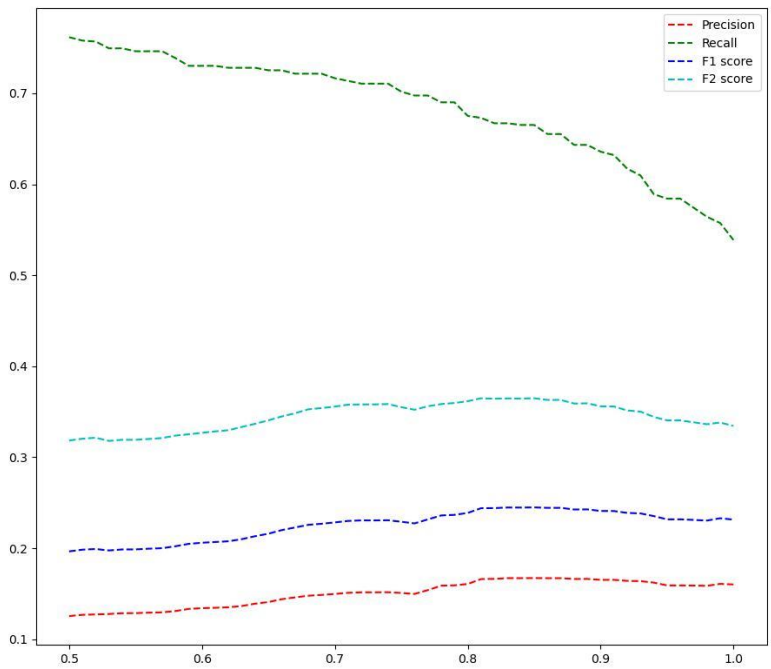


Figure 4.10 The average performances of Min-SDS on different θ values (x-axis). The F2-score peaks at $\theta = 0.85$.

By further analyzing the 3D view of the protein complexes, we can see that the Min-SDS method does have the advantage of predicting unknown hot spots. In the same instance, 1AHW [77], the Min-SDS method predicts 36 residues in the D-E-F area. Of these residues, 18 form hydrogen bonds with residues from another chain (Figure 4.11). To estimate whether a residue is a hot spot, the change in the binding energy from residue mutation is the only metric used. The energy of a hydrogen bond varies from $\approx 5 \sim 6$ kcal/mol for the isolated bond to $\approx 0.5 \sim 1.5$ kcal/mol for proteins in solution [78], close to the threshold 2.0 kcal/mol. When a residue forms a hydrogen bond to another chain, the mutation of this residue will obviously influence the generation of the wild-type hydrogen bond, which should significantly change the binding energy between the chains. Thus, many of the predicted residues in the D-E-F area have the potential to be hot spots.

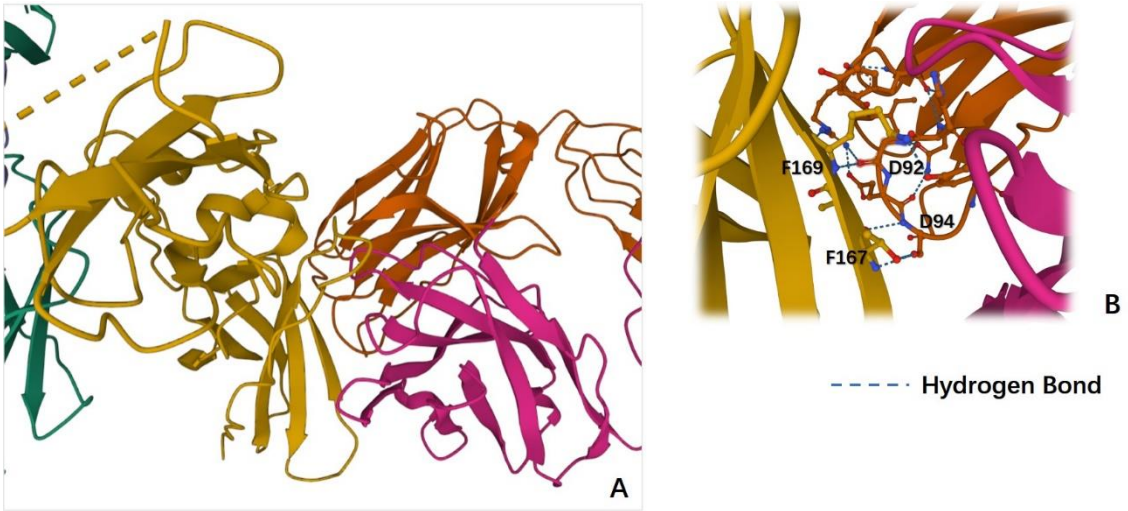


Figure 4.11 A 3D view of the D-E-F area of 1AHW. A: A 3D view of the quaternary structure of 1AHW in the D-E-F area; **B**: Hydrogen bonds between chains D and F.

4.4. Summary and discussion

In this study, we developed three densest subgraph-based methods for protein-protein interaction hot spot prediction. Compared to the existing graph theory-based methods, our methods perform much better in terms of recall and F-score. In particular, our Min-SDS method has an obvious advantage in terms of recall and has the best F2-score among

all the graph theory-based methods. In addition, our Min-DS and Max-DS methods outperform the existing methods in terms of F-score, providing useful network analysis methods for researchers.

Our proposed methods do not consider the weight of edges in the residue interaction network. However, we can use the distance data or solvent-mediated potential data as the weight of edges for further analysis. This requires some new methods which can find densest subgraphs from edge-weighted undirected graphs. Here we propose an LP-based method to find an edge-weighted densest subgraph.

Given an undirected, edge-weighted graph $G = (V, E)$, where the weight of an edge $(i, j) \in E$ is a constant $w_{i,j}$. Let $S = (V', E')$ be a subgraph of G, define the density of S be $\rho(S) = \frac{\left|\sum_{(i,j)\in E'} w_{i,j}\right|}{|V'|}$. If S has the maximum density among all subgraphs of G, then S is a densest subgraph of G.

When all edge weights are positive integers, we can use the following LP method to find an arbitrary densest subgraph.

W-BasicLP:

Maximize $$\sum_{(i,j)\in E} w_{i,j} x_{i,j}$$

Subject to

$$x_{i,j} \le y_i \qquad \forall (i,j) \in E$$

$$x_{i,j} \le y_j \qquad \forall (i,j) \in E$$

$$\sum_{i\in V} y_i \le 1$$

$$x_{i,j} \ge 0, y_i \ge 0 \qquad \forall i, j$$

Proposition 6. *For any subgraph $S = (V', E')$ of G, an optimal solution of W-BasicLP is at least $\rho(S)$.*

Proof: For each $(i, j) \in E'$, set $\overline{x_{i,j}} = \frac{1}{|V'|}$. For each $i \in V'$, set $\overline{y_i} = \frac{1}{|V'|}$. All the remaining variables are set to 0. Then we have, $\sum_{i\in V} \overline{y_i} = |V'| * \frac{1}{|V'|} = 1$. Thus, $(\bar{x}, \bar{y})$

is a feasible solution to the LP. The value of this solution is

$$\sum_{(i,j)\in E'} w_{i,j}\overline{x_{i,j}} = \rho(S)$$

■

Proposition 7. *Given a feasible solution of W-BasicLP with value v, we can construct a subgraph $S = (V', E')$ of G such that $\rho(S) \geq v$.*

Proof: Since $w_{i,j}$ is integer, for each $x_{i,j}$, we can define $w_{i,j}$ additional variables $x_{i,j,k}$, $k \in \{1,2,\dots,w_{i,j}\}$, whose values are equal to $x_{i,j}$. Then the objective function equals to $\sum x_{i,j,k}$.

Let $(\bar{x}, \bar{y})$ be a feasible solution to the LP. Without loss of generality, we can assume that for all ij, $\overline{x_{i,j}} = \min(\bar{y}_i, \bar{y}_j)$.

We define a collection of sets S indexed by a parameter $r \geq 0$. Let $S(r) = \{i | \bar{y}_i \geq r\}$ and $E(r) = \{ijk | \overline{x_{i,j,k}} \geq r\}$. Since $\overline{x_{i,j}} \leq \bar{y}_i$ and $\overline{x_{i,j}} \leq \bar{y}_j$, we have $ijk \in E(r) \Rightarrow i \in V(r), j \in V(r)$. Also, since $\overline{x_{i,j}} = \min(\bar{y}_i, \bar{y}_j)$, we have $i \in V(r), j \in V(r) \Rightarrow ijk \in E(r), \forall k$. Thus, if we ignore the difference of k, $E(r)$ is precisely the set of edges induced by $S(r)$, while each edge ij in $E(r)$ appears $w_{i,j}$ times.

Besides, $\int_0^\infty |S(r)|\, dr = \sum_i \bar{y}_i \leq 1$, and $\int_0^\infty |E(r)|\, dr = \sum_{i,j} w_{i,j}\overline{x_{i,j}}$. See the explanation Figure 4.12.

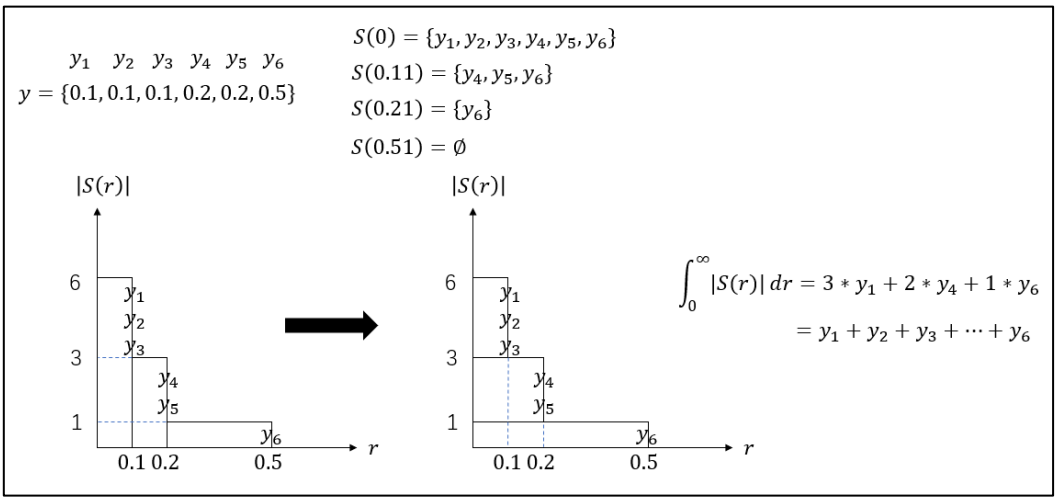


Figure 4.12 An example to explain $\int_0^\infty |S(r)|\, dr = \sum_i \bar{y}_i$. The explanation about $E(r)$ is the similar.

Let $v = \sum_{i,j} w_{i,j}\overline{x_{i,j}}$. We claim that there exists r such that $\frac{|E(r)|}{|S(r)|} \geq v$. Suppose there were no such r. Then

$$v = \int_0^\infty |E(r)|\, dr < v \int_0^\infty |S(r)|\, dr \leq v$$

This gives a contradiction. To find such an r, notice that we can check all combinatorically distinct sets $S(r)$ by simply checking the sets $S(r)$ obtained by setting $r = \overline{y_i}$ for every $i \in V$. Once a feasible r is obtained, we have $\frac{|E(r)|}{|S(r)|} \geq v$, then the subgraph induced by $S(r)$ has $\rho\big(S(r)\big) \geq v$.

■

The proofs of Proposition 6 and Proposition 7 are based on [70]. Combining these two propositions, we can obviously get the following theorem.

Theorem 1. *An optimal solution of* W-BasicLP *leads to an edge-weighted densest subgraph when all the weights are positive integers.*

Because of the linearity in LP, we can multiply a constant number to all $w_{i,j}$'s without changing the ratio of variables in a feasible result of the LP. Thus, for any fractional number-weighted cases, we can use the same LP to solve the edge-weighted densest subgraph problem. In practice, because of the digital precision, any positive-weighted problems can be solved by this LP. The proof of the correctness of the LP under irrational number-weighted condition is left as future work.

Chapter 5

Conclusion and future work

5. Conclusion and future work

In this dissertation, we introduced several graph theory-based methods for solving two bioinformatics problems.

In the first study, we introduced the WMFVS and WFVS methods to predict cancer genes. WMFVS and WFVS take advantage of both bio-data and the network structure. They can be useful in novel cancer gene prediction and evaluation, and the same idea may also be applied to other bioinformatics problems.

The main challenge of our methods is the definition of the weights. WMFVS and WFVS can perform very well when the weights are well-defined but may display limited performance when the weights are not directly related to the category. In this study, we used the gene differential expression data as the weights of genes. Actually, we may use different ideas to define the weights. As an example, since the change of the expression of one gene may influence the other genes, we can use the average (or weighted-average) differential expression data of a certain gene together with its neighbor genes (or even level-2 neighbors, i.e. the neighbors' neighbors) as the weight of the gene. Different definitions of weights should have different results, and finding better definitions of weights is left as future work.

Another issue concerns graph compression. In our experiments, the traditional MFVS method analyzed the compressed graph (with the *ignore* operation; see details in Section 3.2), which contained 660 vertices and 5604 arcs, and it was efficient and took only approximately 4 seconds to obtain the result. The input graph of WMFVS and WFVS was compressed using the limited *ignore_w* operation (see details in Section 3.2), which contained 2348 vertices and 17283 arcs. Because of the different input scales, WMFVS and WFVS were not as efficient as the simple MFVS method, although the time costs were still acceptable. The development of new algorithms for weighted graph compression is left as future work.

We used MFVS as the basic idea in designing our methods, and these methods consider using the features of genes, or vertices in the network, as the weights. There exists another famous concept named minimum feedback arc set (MFCS), which is a set of arcs whose removal leaves the remaining network acyclic. This concept is based on arcs rather than vertices, and we can also apply the MFCS method in predicting cancer genes. In this way, we need to consider the weights of arcs, i.e. the relationships between genes. Constructing new MFCS-based methods and finding appropriate weights of arcs is left as future work.

In the second study, we proposed three densest subgraph-based methods to predict the hot spot residues in protein complexes, including Min-DS, Max-DS, and Min-SDS. We also implemented two existing graph theory-based methods, mincut and biclique, for comparison. Although the Mincut method has the best precision, its predictions tend to be concentrated in one connected subgraph, which significantly reduces the recall in practice. In comparison, the results of our DS-based methods are not restricted to one connected component, which is important in dealing with complexes with multiple binding sites.

Compared to machine learning methods, our DS-based methods do not depend on insufficient bio-experimental data and thus have the advantage of being able to search unknown hot spots without many data resources. Our DS-based methods use only spatial coordinate information to detect important vertices in a given interaction network. The high recall scores make them good choices for some other high-false-negative-rate network analyses, and they can be easily applied to various network analysis fields.

In the original Biclique methods, researchers first find all the bicliques as the hot spot patterns and then search the frequently appeared patterns among all protein complex data. In the Min-DS and Min-SDS methods, usually, multiple residue groups are detected. We may also treat these groups as patterns and compare them to the other patterns from different complexes for further selection. However, since the found residue groups in the Min-DS and Min-SDS are significantly larger than the patterns in Biclique, we may need some fuzzy matching methods for pattern comparison. Based on this idea, we can

consider using the hot spot information from other complexes to improve the precision of the results, and this is left as future work.

In Section 4.4, we proposed a basic idea about the edge-weighted densest subgraph method to analysis the residue interaction network. However, the design of more practical methods and finding an appropriate definition of the weights of edges is left as another future work.

Reference

[1] W. Gao, H. Wu, M. K. Siddiqui, and A. Q. Baig, "Study of biological networks using graph theory," *Saudi J Biol Sci,* vol. 25, no. 6, pp. 1212-1219, 2018.

[2] J. W. Heal, G. J. Bartlett, C. W. Wood, A. R. Thomson, and D. N. Woolfson, "Applying graph theory to protein structures: an Atlas of coiled coils," *Bioinformatics,* vol. 34, no. 19, pp. 3316-3323, 2018.

[3] X. Qi, Q. Wu, Y. Zhang, E. Fuller, and C.-Q. Zhang, "A novel model for DNA sequence similarity analysis based on graph theory," *Evol Bioinf,* vol. 7, pp. EBO-S7364, 2011.

[4] N. J. da Fonseca Jr, M. Q. L. Afonso, L. C. de Oliveira, and L. Bleicher, "A new method bridging graph theory and residue co-evolutionary networks for specificity determinant positions detection," *Bioinformatics,* vol. 35, no. 9, pp. 1478-1485, 2019.

[5] E. Tolan and Z. Isik, "Graph theory based classification of brain connectivity network for autism spectrum disorder," in *International Conference on Bioinformatics and Biomedical Engineering*, 2018, pp. 520-530.

[6] M. Babi, J. Miheli, and M. Cal, "Complex network characterization using graph theory and fractal geometry: The case study of lung cancer DNA sequences," *Appl Sci,* vol. 10, no. 9, p. 3037, 2020.

[7] J. R. Boes, O. Mamun, K. Winther, and T. Bligaard, "Graph theory approach to high-throughput surface adsorption structure generation," *J Phys Chem A,* vol. 123, no. 11, pp. 2281-2285, 2019.

[8] S. Manganas, N. Bourbakis, and K. Michalopoulos, "Brain structural and functional representation based on the local global graph methodology," in *2018 IEEE 18th International Conference on Bioinformatics and Bioengineering (BIBE)*, 2018, pp. 139-142.

[9] R. Gonzlez-Alemn *et al.*, "BitQT: a graph-based approach to the quality threshold clustering of molecular dynamics," *Bioinformatics,* vol. 38, no. 1, pp. 73-79, 2021.

[10] M. Habibi, C. Eslahchi, M. Sadeghi, and H. Pezashk, "The interpretation of protein structures based on graph theory and contact map," *Open Access Bioinf,* vol. 2, pp. 127-137, 2010.

[11] H. N. Gabow, "Path-Based Depth-First Search for Strong and Biconnected Components," *Inf Process Lett,* vol. 74, pp. 107-114, 2000.

[12] L. Gurobi Optimization. "Gurobi optimizer reference manual." http://www.gurobi.com.

[13] K. S. Thorn and A. A. Bogan, "ASEdb: a database of alanine mutations and their effects on the free energy of binding in protein interactions," *Bioinformatics,* vol.

17, no. 3, pp. 284-285, 2001.
[14] S. K. Burley *et al.*, "RCSB Protein Data Bank: powerful new tools for exploring 3D structures of biological macromolecules for basic and applied research and education in fundamental biology, biomedicine, biotechnology, bioengineering and energy sciences," *Nucleic Acids Res,* vol. 49, no. D1, pp. D437-D451, 2021.
[15] P. K. Vogt, "Cancer genes," *West J Med,* vol. 158, no. 3, pp. 273-278, 1993.
[16] P. Luo, Y. Ding, X. Lei, and F. X. Wu, "deepDriver: predicting cancer driver genes based on somatic mutations using deep convolutional neural networks," *Front Genet,* vol. 10, p. 13, 2019.
[17] C. J. Tokheim, N. Papadopoulos, K. W. Kinzler, B. Vogelstein, and R. Karchin, "Evaluating the evaluation of cancer driver genes," *Proc Natl Acad Sci,* vol. 113, no. 50, pp. 14330-14335, 2016.
[18] G. Parmigiani, S. Boca, J. Lin, K. W. Kinzler, V. Velculescu, and B. Vogelstein, "Design and analysis issues in genome-wide somatic mutation studies of cancer," *Genomics,* vol. 93, no. 1, p. 17, 2009.
[19] F. Cheng, J. Zhao, and Z. Zhao, "Advances in computational approaches for prioritizing driver mutations and significantly mutated genes in cancer genomes," *Briefings Bioinf,* vol. 17, no. 4, pp. 642-656, 2016.
[20] H. Dai, L. Li, T. Zeng, and L. Chen, "Cell-specific network constructed by single-cell RNA sequencing data," *Nucleic Acids Res,* vol. 47, no. 11, pp. e62-e62, 2019.
[21] M. T. W. Ebbert *et al.*, "Systematic analysis of dark and camouflaged genes reveals disease-relevant genes hiding in plain sight," *Genome Biol,* vol. 20, no. 1, p. 97, 2019.
[22] J. G. T. Zaudo, G. Yang, and R. Albert, "Structure-based control of complex networks with nonlinear dynamics," *Proc Natl Acad Sci,* vol. 114, no. 28, pp. 7234-7239, 2017.
[23] A. Mochizuki, B. Fiedler, G. Kurosawa, and D. Saito, "Dynamics and control at feedback vertex sets. II: A faithful monitor to determine the diversity of molecular activities in regulatory networks," *J Theor Biol,* vol. 335, pp. 130-146, 2013.
[24] W. F. Guo *et al.*, "Discovering personalized driver mutation profiles of single samples in cancer by network control strategy," *Bioinformatics,* vol. 34, no. 11, pp. 1893-1903, 2018.
[25] Y. Bao, M. Hayashida, P. Liu, M. Ishitsuka, J. C. Nacher, and T. Akutsu, "Analysis of critical and redundant vertices in controlling directed complex networks using feedback vertex sets," *J Comput Biol,* vol. 25, no. 10, pp. 1071-1090, 2018.
[26] W. Zheng, D. Wang, and X. Zou, "Control of multilayer biological networks and applied to target identification of complex diseases," *BMC Bioinf,* vol. 20, no. 1, pp. 1-12, 2019.
[27] M. R. Garey and D. S. Johnson, *Computers and Intractability*. San Francisco:

Freeman, 1979.
[28] V. Guruswami and E. Lee, "Inapproximability of Feedback Vertex Set for Bounded Length Cycles," in *Electronic Colloquium on Computational Complexity (ECCC)*, 2014, vol. 21, p. 2.
[29] A. Becker, R. Bar-Yehuda, and D. Geiger, "Randomized algorithms for the loop cutset problem," *J Artif Intell Res,* vol. 12, pp. 219-234, 2000.
[30] Y. Cao, J. Chen, and Y. Liu, "On feedback vertex set: New measure and new structures," *Algorithmica,* vol. 73, no. 1, pp. 63-86, 2015.
[31] F. V. Fomin and Y. Villanger, "Finding induced subgraphs via minimal triangulations," *arXiv preprint arXiv:0909.5278,* 2009.
[32] S. T. Chakradhar, A. Balakrishnan, and V. D. Agrawal, "An exact algorithm for selecting partial scan flip-flops," *J Electron Test,* vol. 7, no. 1-2, pp. 83-93, 1995.
[33] B. Schwikowski and E. Speckenmeyer, "On enumerating all minimal solutions of feedback problems," *Discrete Appl Math,* vol. 117, no. 1-3, pp. 253-265, 2002.
[34] E. L. Lloyd, M. L. Soffa, and C. C. Wang, "On locating minimum feedback vertex sets," *J Comput Syst Sci,* vol. 37, no. 3, pp. 292-311, 1988.
[35] G. Smith and R. Walford, "The identification of a minimal feedback vertex set of a directed graph," *IEEE Trans Circuits Syst,* vol. 22, no. 1, pp. 9-15, 1975.
[36] W. Jiang, T. Liu, T. Ren, and K. Xu, "Two hardness results on feedback vertex sets," in *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*. Berlin, Heidelberg: Springer, 2011, pp. 233-243.
[37] F. R. Madelaine and I. A. Stewart, "Improved upper and lower bounds on the feedback vertex numbers of grids and butterflies," *Discrete Math,* vol. 308, no. 18, pp. 4144-4164, 2008.
[38] A. Vinayagam *et al.*, "A directed protein interaction network for investigating intracellular signal transduction," *Sci Signaling,* vol. 4, no. 189, pp. rs8-rs8, 2011.
[39] Y. Liu, J. Sun, and M. Zhao, "ONGene: a literature-based database for human oncogenes," *J Genet Genomics,* vol. 44, no. 2, pp. 119-121, 2017.
[40] M. Zhao, J. Sun, and Z. Zhao, "TSGene: a web resource for tumor suppressor genes," *Nucleic Acids Res,* vol. 41, no. D1, pp. D970-D976, 2013.
[41] Z. Sondka, S. Bamford, C. G. Cole, S. A. Ward, I. Dunham, and S. A. Forbes, "The COSMIC Cancer Gene Census: describing genetic dysfunction across all human cancers," *Nat Rev Cancer,* vol. 18, no. 11, pp. 696-705, 2018.
[42] D. Repana *et al.*, "The Network of Cancer Genes (NCG): a comprehensive catalogue of known and candidate cancer genes from cancer sequencing screens," *Genome Biol,* vol. 20, no. 1, p. 1, 2019.
[43] A. Subramanian *et al.*, "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles," *Proc Natl Acad Sci,* vol. 102, no. 43, pp. 15545-15550, 2005.
[44] J. N. Weinstein *et al.*, "The cancer genome atlas pan-cancer analysis project,"

Nat Genet, vol. 45, no. 10, p. 1113, 2013.
[45] C. Y. Lin *et al.*, "Membrane protein-regulated networks across human cancers," *Nat Commun,* vol. 10, no. 1, pp. 1-17, 2019.
[46] D. O. Morgan, *The Cell Cycle: Principles of Control*. London: New Science Press, 2007.
[47] S. W. Chae *et al.*, "Overexpressions of Cyclin B1, cdc2, p16 and p53 in human breast cancer: the clinicopathologic correlations and prognostic implications," *Yonsei Med J,* vol. 52, no. 3, pp. 445-453, 2011.
[48] A. Mao *et al.*, "ZBTB7A promotes migration, invasion and metastasis of human breast cancer cells through NF-κB-induced epithelial–mesenchymal transition in vitro and in vivo," *J Biochem,* vol. 166, no. 6, pp. 485-493, 2019.
[49] J. De Las Rivas and C. Fontanillo, "Protein--protein interactions essentials: key concepts to building and analyzing interactome networks," *PLoS Comput Biol,* vol. 6, no. 6, p. e1000807, 2010.
[50] A. A. Bogan and K. S. Thorn, "Anatomy of hot spots in protein interfaces," *J Mol Biol,* vol. 280, no. 1, pp. 1-9, 1998.
[51] N. Tuncbag, A. Gursoy, and O. Keskin, "Identification of computational hot spots in protein interfaces: combining solvent accessibility and inter-residue potentials improves the accuracy," *Bioinformatics,* vol. 25, no. 12, pp. 1513-1520, 2009.
[52] T. B. Fischer *et al.*, "The binding interface database (BID): a compilation of amino acid hot spots in protein interfaces," *Bioinformatics,* vol. 19, no. 11, pp. 1453-1454, 2003.
[53] J. Jankauskait, B. Jimnez-Garca, J. Dapknas, J. Fernndez-Recio, and I. H. Moal, "SKEMPI 2.0: an updated benchmark of changes in protein--protein binding energy, kinetics and thermodynamics upon mutation," *Bioinformatics,* vol. 35, no. 3, pp. 462-469, 2019.
[54] I. S. Moreira, P. A. Fernandes, and M. J. Ramos, "Hot spots—A review of the protein--protein interface determinant amino-acid residues," *Proteins: Struct, Funct, Bioinf,* vol. 68, no. 4, pp. 803-812, 2007.
[55] I. Massova and P. A. Kollman, "Computational alanine scanning to probe protein-protein interactions: a novel approach to evaluate binding free energies," *J Am Chem Soc,* vol. 121, no. 36, pp. 8133-8143, 1999.
[56] S. Grosdidier and J. Fernndez-Recio, "Identification of hot-spot residues in protein-protein interactions by computational docking," *BMC Bioinf,* vol. 9, no. 1, pp. 1-13, 2008.
[57] R. Guerois, J. E. Nielsen, and L. Serrano, "Predicting changes in the stability of proteins and protein complexes: a study of more than 1000 mutations," *J Mol Biol,* vol. 320, no. 2, pp. 369-387, 2002.
[58] T. Kortemme, D. E. Kim, and D. Baker, "Computational alanine scanning of protein-protein interfaces," *Sci STKE,* vol. 2004, no. 219, pp. pl2-pl2, 2004.
[59] H. Wang, C. Liu, and L. Deng, "Enhanced prediction of hot spots at protein-

protein interfaces using extreme gradient boosting," *Sci Rep,* vol. 8, no. 1, pp. 1-13, 2018.

[60] L. Wang, Z.-P. Liu, X.-S. Zhang, and L. Chen, "Prediction of hot spots in protein interfaces using a random forest model with hybrid features," *Protein Eng, Des Sel,* vol. 25, no. 3, pp. 119-126, 2012.

[61] J.-F. Xia, X.-M. Zhao, J. Song, and D.-S. Huang, "APIS: accurate prediction of hot spots in protein interfaces by combining protrusion index with solvent accessibility," *BMC Bioinf,* vol. 11, no. 1, pp. 1-14, 2010.

[62] Q. Liu, P. Chen, B. Wang, J. Zhang, and J. Li, "Hot spot prediction in protein-protein interactions by an ensemble system," *BMC Syst Biol,* vol. 12, no. 9, pp. 89-99, 2018.

[63] L. Liqi, K. Hong, Z. Yuan, Z. Yue, W. Kaifa, and W. Ying, "Prediction of eukaryotic protein subcellular multi-localisation with a combined KNN-SVM ensemble classifier," *J Comput Biol Bioinf Res,* vol. 3, no. 2, pp. 15-24, 2011.

[64] Y. Gao, R. Wang, and L. Lai, "Structure-based method for analyzing protein--protein interfaces," *J Mol Model,* vol. 10, no. 1, pp. 44-54, 2004.

[65] N. Tuncbag, F. S. Salman, O. Keskin, and A. Gursoy, "Analysis and network representation of hotspots in protein interfaces using minimum cut trees," *Proteins: Struct, Funct, Bioinf,* vol. 78, no. 10, pp. 2283-2294, 2010.

[66] J. Li and Q. Liu, "`Double water exclusion': a hypothesis refining the O-ring theory for the hot spots at protein interfaces," *Bioinformatics,* vol. 25, no. 6, pp. 743-750, 2009.

[67] O. D. Balalau, F. Bonchi, T. H. H. Chan, F. Gullo, and M. Sozio, "Finding subgraphs with maximum total density and limited overlap," in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 2015, pp. 379-388.

[68] L. Qin, R.-H. Li, L. Chang, and C. Zhang, "Locally densest subgraph discovery," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 965-974.

[69] D. Lim *et al.*, "Crystal structure and kinetic analysis of β-lactamase inhibitor protein-II in complex with TEM-1 β-lactamase," *Nat Struct Biol,* vol. 8, no. 10, pp. 848-852, 2001.

[70] M. Charikar, "Greedy approximation algorithms for finding dense components in a graph," in *International Workshop on Approximation Algorithms for Combinatorial Optimization*, 2000, pp. 84-95.

[71] L. R. Ford Jr and D. R. Fulkerson, *Flows in Networks*. Princeton university press, 2015.

[72] R. E. Gomory and T. C. Hu, "Multi-terminal network flows," *J Soc Ind Appl Math,* vol. 9, no. 4, pp. 551-570, 1961.

[73] E. S. Ozdemir, A. Gursoy, and O. Keskin, "Analysis of single amino acid variations in singlet hot spots of protein--protein interfaces," *Bioinformatics,* vol. 34, no. 17, pp. i795-i801, 2018.

[74] P. Mitra and D. Pal, "Combining Bayes classification and point group symmetry under Boolean framework for enhanced protein quaternary structure inference," *Struct,* vol. 19, no. 3, pp. 304-312, 2011.
[75] E. Krissinel, "Crystal contacts as nature's docking solutions," *J Comput Chem,* vol. 31, no. 1, pp. 133-143, 2010.
[76] S. Sirin, J. R. Apgar, E. M. Bennett, and A. E. Keating, "AB-Bind: antibody binding mutational database for computational affinity predictions," *Protein Sci,* vol. 25, no. 2, pp. 393-409, 2016.
[77] M. Huang *et al.*, "The mechanism of an inhibitory antibody on TF-initiated blood coagulation revealed by the crystal structures of human tissue factor, Fab 5G9 and TF· 5G9 complex," *J Mol Biol,* vol. 275, no. 5, pp. 873-894, 1998.
[78] S.-Y. Sheu, D.-Y. Yang, H. L. Selzle, and E. W. Schlag, "Energetics of hydrogen bonds in peptides," *Proc Natl Acad Sci,* vol. 100, no. 22, pp. 12683-12687, 2003.

Publication list

***Chapter 3* is based on a paper in *BMC Bioinformatics*:**

Li, R., Lin, C. Y., Guo, W. F., & Akutsu, T. (2021). Weighted minimum feedback vertex sets and implementation in human cancer genes detection. *BMC bioinformatics*, 22(1), 1-17.

***Chapter 4* is based on a paper in *BMC Bioinformatics*:**

Li, R., Lee, J., Yang, J., & Akutsu, T. (2022). Densest subgraph-based methods for protein-protein interaction hot spot prediction. *BMC bioinformatics*, 23, 451.

Other publications

(1) Akutsu, T., Jansson, J., Li, R., Takasu, A., & Tamura, T. (2018). New and Improved Algorithms for Unordered Tree Inclusion. In *29th International Symposium on Algorithms and Computation* (*ISAAC 2018*), 27: 1-12.

Akutsu, T., Jansson, J., Li, R., Takasu, A., & Tamura, T. (2021). New and improved algorithms for unordered tree inclusion. *Theoretical Computer Science*, 883, 83-98.

(2) Lin, C. Y., Ruan, P., Li, R., Yang, J. M., See, S., Song, J., & Akutsu, T. (2019). Deep learning with evolutionary and genomic profiles for identifying cancer subtypes. *Journal of Bioinformatics and Computational Biology*, 17(03), 1940005.

Lin, C. Y., Ruan, P., Li, R., Yang, J. M., See, S. & Akutsu, T. (2018). Deep Learning with Evolutionary and Genomic Profiles for Identifying Cancer Subtypes. In *2018 IEEE 18th International Conference on Bioinformatics and Bioengineering (BIBE)*. pp. 147-150.