DISCOVERING COMMUNITY STRUCTURE BY OPTIMIZING COMMUNITY QUALITY METRICS

By

Mingming Chen

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: COMPUTER SCIENCE

Approved by the
Examining Committee:

Boleslaw K. Szymanski, Thesis Adviser

Elliot Anshelevich, Member

Gyorgy Korniss, Member

Sibel Adali, Member

Rensselaer Polytechnic Institute
Troy, New York

November 2015
(For Graduation December 2015)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

Firstly, I would like to express my deepest gratitude to my advisor Prof. Boleslaw K. Szymanski for his continuous support, guidance, and encouragement through the course of my degree. His guidance helps me in all the time of research and writing of this thesis. I am very lucky to have him as my advisor and it is really happy to work with him.

Besides my advisor, I would like to thank the rest of my thesis committee, Prof. Elliot Anshelevich, Prof. Gyorgy Korniss, and Prof. Sibel Adali, for their valuable suggestion and comments on my candidacy report, thesis, and presentation. In addition, I would like to especially thank Prof. Mark K. Goldberg who served as a committee member of my candidacy exam and research qualifying exam. Although he is not in my thesis committee due to the reason of his retirement from RPI, he still read my thesis carefully and gave me valuable feedback. He always makes the decision that is good for me which I appreciate a lot.

I would like to thank my colleagues, Konstantin Kuzmin and Ashwin Bahulkar. I really had a great time working with them and I learnt a lot from them. Konstantin did an excellent job in writing the summary for resolution limit problem of modularity (Section 2.2.3), for resolving resolution limit problem of modularity (Section 2.2.4), and for link prediction based on link prediction metrics and community structure (Section 6.1). I also thank him for his help with writing for the link replacing methodology (Section 6.2). Ashwin helped a lot in running the experiments to generate the modified networks by using the link replacing methodology described in Section 6.2. I would also like to thank Thomas Babbitt for his help with the rehearsals for my thesis defense. Without his suggestion and comments on the slides and presentation, the defense cannot be as successful as it was.

Finally, I would like to thank my collaborators, Chris Gaiteri from Rush University Medical Center and Albert Trias Mansilla from University of Girona. Working with both of them broadens my research horizon.

This work was supported in part by the Army Research Laboratory under

Cooperative Agreement Number W911NF-09-2-0053 and by the the Office of Naval Research Grant No. N00014-09-1-0607. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies either expressed or implied of the Army Research Laboratory or the U.S. Government.

ABSTRACT

Many networks contain community structure which identifies groups of nodes within which connections are denser than between them. Detecting and characterizing such community structure, which is known as community detection, is one of the fundamental issues in the study of network systems. It has received a considerable attention in the last years. Numerous techniques have been developed for both efficient and effective community detection. The most popular one has been to maximize the community quality metric known as Newman's modularity over all the possible partitions of a network. This metric measures the difference between the fraction of all edges that are within the actual community and a fraction of such edges in a randomized graph with the same number of nodes and the same degree sequence. It is widely used to measure the strength of the community structure detected by the community detection algorithms.

However, modularity maximization suffers from two opposite yet concurrent problems. In some cases, it tends to split large communities into smaller communities. In other cases, it tends to form large communities by merging communities that are smaller than a certain threshold which depends on the total number of edges in the network and on the degree of inter-connectivity between the communities. The latter problem is well-known in the literature as the resolution limit problem. To solve these two problems simultaneously, we propose a new community quality metric, that we termed *Modularity Density*, as an alternative to modularity. First, we show modularity decreased by *Split Penalty*, defined as the fraction of edges that connect nodes of different communities, resolves the issue of favoring small communities. Then, we demonstrate that including community densities into modularity and split penalty eliminates the problem of favoring large communities, namely the resolution limit problem.

In addition, modularity can only be used to quantify the quality of disjoint communities. However, it is more realistic to expect that nodes in real-world networks belong to more than one community, resulting in overlapping communities.

In the past few years, several overlapping extensions of modularity were proposed to measure the quality of overlapping community structure. However, all these extensions differ just in the way they define the belonging coefficient and belonging function. Yet, there is lack of systematic comparison of different extensions. To fill this gap, we overview overlapping extensions of modularity and generalize them with a uniform definition enabling application of different belonging coefficients and belonging functions to select the best. In addition, we extend localized modularity, modularity density, and eight local community quality metrics to enable their usages for overlapping communities.

We then propose a novel fine-tuned disjoint community detection algorithm that repeatedly attempts to improve the quality metrics by splitting and merging the given community structure. This new algorithm can actually be used to optimize any community quality metric. However, in this thesis, we only consider modularity and modularity density.

Although community detection is one of the fundamental techniques of network science, the community structure of networks discovered by community detection algorithms does not usually represent the reality. The primary reason for this is incompleteness and inaccuracy of current network data collection methods, which may cause datasets to appear less modular than the underlying networks really are. Thus, in this thesis we aim at recovering or improving the network community structure which may be hidden or impaired because of the missing or incorrectly identified extraneous edges. To this end, we introduce a method for improving the network structure. This method uses the scores obtained from different link prediction techniques to replace a certain fraction of low ranking existing links with the top ranked predicted links.

CHAPTER 1
INTRODUCTION

Communities are the basic structures in sociology in general and in social networks in particular. They have been intensively researched for more than a half of the century [1]. Community in sociology usually refers to a social unit whose members share common values and the identity of the members as well as their degree of cohesiveness depend on individuals' social and cognitive factors such as beliefs, preferences, or needs. The ubiquity of the Internet and social media eliminated spatial limitations on community geographical range, enabling on-line communities to link people regardless of their physical location. The newly arising computational sociology relies on computationally intensive methods to analyze and model social phenomena [2], including communities and their detection.

Analysis of social networks became one of the basic tools of sociology [3] and has been used for linking micro and macro levels of sociological theory. The classical example of the approach is presented in [4] that elaborated the macro implications of one aspect of small-scale interaction, the strength of dyadic ties. Moreover, a lot of commercial applications, such as digital marketing, behavioral targeting, and user preference mining, rely heavily on community analysis. With the rapid growth of large-scale on-line social networks, e.g., Facebook connected a billion users in 2012, there is a high demand for efficient community detection algorithms that will be able to handle their evolutionary growth. Communities in on-line social networks are discovered by analyzing the observed and often recorded on-line interactions

Portions of this chapter previously appeared as: M. Chen, T. Nguyen, and B. K. Szymanski, "A new metric for quality of network community structure," *ASE Human J.*, vol. 2, no. 4, pp. 226-240, Sep. 2013.

Portions of this chapter previously appeared as: M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE T. Comput. Soc. Syst.*, vol. 1, no. 1, pp. 46-65, Mar. 2014.

Portions of this chapter previously appeared as: M. Chen and B. K. Szymanski, "Fuzzy overlapping community quality metrics," *Soc. Netw. Anal. Min.*, vol. 5, no. 1, pp. 1-14, Jul. 2015.

Portions of this chapter have been submitted as: M. Chen, A. Bahulkar, K. Kuzmin, and B. K. Szymanski, "Improving network community structure with link prediction ranking," in *Proc. 7th Workshop Complex Networks* (under review), 2016.

between people.

In computational sociology, communities are defined as groups of nodes in a social network within which connections are denser than between them [5]. This definition has been found useful also in other type of networks, and community detection became one of the fundamental issues in network science. Community detection has been shown to reveal latent yet meaningful structure not only for groups in online and contact-based social networks, but also in groups of customers with similar interests in online retailer user networks, groups of scientists in interdisciplinary collaboration networks, and in biology in functional modules in protein-protein interaction networks etc. [6].

Since in most applications the real communities are not known (often due to the cost of establishing ground truth in large on-line social networks), there is a need for developing reliable metrics to evaluate detected communities, so these metrics can be used to rank the quality of community structure discovered by different community detection algorithms. Such metrics can also be used to develop novel community detection algorithms that iteratively attempt to improve the metrics by merging or splitting the given network community structure.

In the last decade, the most popular community detection method, proposed by Newman [7], has been to maximize the quality metric known as modularity [5, 8–10] over all the possible partitions of a network. It measures the difference (relative to the total number of edges) between the actual and expected (in a randomized graph with the same number of nodes and the same degree distribution) number of edges within a given community. It is widely used to measure the strength of the community structure detected by the community detection algorithms. However, modularity maximization has two opposite yet coexisting problems. In some cases, it favors small communities by splitting large communities. In other cases, it favors large communities by failing to discover communities smaller than a certain threshold even when such communities are well defined. The threshold depends on the total number of edges in the network and on the degree of inter-connectedness between the communities. The latter problem is well-known in the literature as the resolution limit problem [11].

To solve these two problems simultaneously, we propose a new community quality metric, that we termed modularity density, as an alternative to modularity. First, we show modularity decreased by split penalty, defined as the fraction of edges that connect nodes of different communities, resolves the issue of favoring small communities. Next, we show that including community densities into modularity and split penalty eliminates the problem of favoring large communities, namely the resolution limit problem. We demonstrate with proofs and experiments on real-world dynamic datasets that modularity density is an effective alternative to modularity.

In addition, Newman's modularity [5, 8–10] can only be used to measure the quality of disjoint communities. However, it is more realistic to expect that nodes in real-world networks belong to more than one community, resulting in overlapping communities [12]. For instance, a researcher may be active in several research areas, and a node in biological networks might have multiple functions. It is also quite common that people in social networks are naturally characterized by multiple community memberships depending on their families, friends, professional colleagues, neighbors, etc. For this reason, discovering overlapping communities became very popular in the last few years. Several overlapping extensions of modularity ([13–19]) were proposed to measure the quality of overlapping community structure. Yet, to date no attempt has been made to systematically compare different overlapping extensions and propose metric selection criteria for different types of networks. Consequently, we consider several overlapping extensions of modularity and test their quality on real-world and synthetic networks. We also extend localized modularity [20], modularity density [21, 22], and eight local community quality metrics for overlapping communities following the same principles used by the overlapping extensions of modularity.

We conducted experiments on a large number of real-world networks and synthetic networks using overlapping extensions of modularity, overlapping modularity density, and eight local metrics. The results show that selecting the product of the belonging coefficients of two nodes as a belonging function for overlapping extensions yields better results on these networks than using other belonging functions. The experimental results also give a guidance to researchers on which metrics to

choose when measuring the quality of overlapping community structure.

We also propose a novel fine-tuned disjoint community detection algorithm that repeatedly attempts to improve the quality metrics by splitting and merging the given community structure. We denote the corresponding algorithm based on modularity (Q) as *Fine-tuned* Q while the one based on modularity density (Q_{ds}) is referred to as *Fine-tuned* Q_{ds}. This new algorithm can actually be used to optimize any community quality metric. However, we only consider modularity and modularity density in this thesis.

Although community detection is one of the fundamental issues in network systems and it is expected that many networks, like social and biology networks, have highly modular subsets or, in other words, community structure, the community structure discovered by community detection algorithms does not usually represent the reality. The primary reason for this is that available network datasets are often incomplete and inaccurate. For example, in the process of collecting, gathering, or recording information from online social networks, some data can be lost or incorrect because of complex relations between individuals, privacy constraints, improper or imprecise sampling methods, etc. Also, in a network representing interactions between genes in some species edges are typically determined experimentally, so the number of known edges may be much smaller than in reality. Moreover, random spatial collocation of some genes may be wrongly interpreted as an active interaction. Consequently, the networks we derive from available data usually have some noise, like missing some edges or having some incorrectly identified so extraneous edges, which may cause the collected datasets to appear less modular than the underlying networks really are.

Thus, the purpose of this thesis is to propose and evaluate methods of recovering or improving the network community structure which may be hidden or impaired by missing or extraneous edges. Our goal is to make the network more modular by recovering missing edges and removing extraneous edges. We introduce a link improvement procedure that removes a certain fraction of existing low ranking links and replaces them with potential links (e.g., the links of the complete graph with the same set of nodes as the current graph that do not exist in the current

graph) ranked highly by a link prediction metric. The proposed method is able to significantly improve the community structure of the networks we considered.

1.1 Contributions and Organization

1.1.1 Contributions

The contributions of this thesis are as follows:

- This thesis first introduces a new community quality metric, called modularity density that resolves two well-known issues of modularity, for quantifying the quality of network community structure. We show that in many cases in which modularity suffers from limitations while our modularity density does not.

- We overview overlapping extensions of modularity and generalize them with a uniform definition enabling application of different belonging coefficients and belonging functions to select the best. In addition, we extend localized modularity, modularity density, and eight local community quality metrics to enable their usages for overlapping communities.

- We then propose a novel fine-tuned disjoint community detection algorithm that repeatedly attempts to improve the community quality metrics by splitting and merging the given community structure.

- We introduce an approach to improve the network community structure by removing a certain fraction of low ranking existing links and replacing them with highly ranked predicted links.

1.1.2 Organization

The structure of this thesis is as follows. Chapter 2 offers a literature review about the definitions of community structure, the definition of modularity, the community detection algorithms based on maximizing modularity, the well-known problems of modularity, and the solutions to resolve these problems. Chapter 3 introduces a new metric, called modularity density, for measuring the quality of network community structure. Modularity density eliminates the two well-known issues of

modularity simultaneously. In Chapter 4, we overview overlapping extensions of modularity and generalize them with a uniform definition enabling application of different belonging coefficients and belonging functions to select the best. We also extend localized modularity, modularity density, and eight local community quality metrics to enable their usages for overlapping community structure. Chapter 5 introduces a novel fine-tuned disjoint community detection algorithm that repeatedly attempts to improve the quality metrics by splitting and merging the given community structure. We then introduce an link improvement approach to improve the network community structure by removing a certain fraction of low ranking existing links and replacing them with highly ranked predicted links in Chapter 6. Finally, we conclude and discuss the future work in Chapter 7.

CHAPTER 2
LITERATURE REVIEW

2.1 Community Structure

Community structure is a common feature to many networks, including Internet, citation networks, transportation networks, email networks, and social and biochemical networks. It has attracted a great deal of interest recently. However, there is no single, universally accepted definition of a community within a social network. One popular definition is that a community is a collection of nodes more strongly connected than would occur from random chance, leading to the definition of modularity [5].

Community can be defined in a very strict sense as clique which is a complete connected subgraph. A k-clique community is a subset of k nodes that are adjacent to each other. However, this definition is so strong that it is rarely fulfilled in real sparse networks for larger groups [23]. Community can also be defined as k-core. A k-core is a subgraph in which each node is adjacent to at least k other nodes of the subgraph [24]. It is weaker than a clique.

Radicchi et al. [25] proposed a quantitative definition for community which is a subset of nodes of the network such that connections between them are denser than connections with the rest of the network. In a strong sense, a subgraph c of a whole graph $G = (V, E)$ is a community if

$$k_i^{in} > k_i^{out}, \forall i \in c, \tag{2.1}$$

where k_i^{in} and k_i^{out} are the in- and out- degrees of node i. In a strong community, each node has more connections inside its community than with the rest of the network. It coincides with the one proposed in [26] in the framework of the identification of web communities. This strong definition is also similar to the definition of LS-set

Portions of this chapter previously appeared as: M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE T. Comput. Soc. Syst.*, vol. 1, no. 1, pp. 46-65, Mar. 2014.

[3], although LS-set is much more stringent. The LS-set is defined as a set of nodes in which each of its subsets has more connections to its components within the set than outside. In a weak sense, c is a community if

$$2|E_c^{in}| > |E_c^{out}|, \tag{2.2}$$

where $|E_c^{in}|$ is the number of edges between nodes within community c and $|E_c^{out}|$ is the number of edges from the nodes in community c to the nodes outside c. In a weak community, the sum of all degrees within c is larger than the sum of all degrees toward the rest of the network. It is clear that a community in a strong sense is also a community in a weak sense, whereas the converse is not true.

Hu et al. [27] introduced alternative definitions for strong and weak communities. Community c is a strong community if the internal degree of any node in c is larger than or at least equal to the number of edges that this node shares with any other community. That is,

$$k_i^c \geq \max_{c' \in C, c' \neq c} k_i^{c'}, \forall i \in c, \tag{2.3}$$

where k_i^c is the internal degree of node i in c, $k_i^{c'}$ is the number of edges between node i and community c', and C is the set of communities. Community c is a weak community if the number of internal edges of c is larger than or at least equal to the number of edges shared by this community with any other community. Namely,

$$|E_c^{in}| \geq \max_{c' \in C, c' \neq c} |E_{c,c'}|. \tag{2.4}$$

where $|E_{c,c'}|$ is the number of edges from community c to community c'. Note that the strong community defined in Equation (2.3) is weaker than that defined in Equation (2.1) and the weak community defined in Equation (2.4) is also weaker than that defined in Equation (2.2).

Li, Zhang et al. [28] stated that the average modularity degree $d(c)$ of a

community c

$$\begin{aligned} d(c) &= d_{in}(c) - d_{out}(c) \\ &= \frac{2\,|E_c^{in}| - |E_c^{out}|}{|c|}, \end{aligned} \tag{2.5}$$

should be as large as possible for c to be a "valid" community. In the formula, $d_{in}(c)$ is the average inner degree of c (also known as *contraction*) and $d_{out}(c)$ is the average outer degree of c (also known as *expansion*).

Mancoridis et al. [29] suggested that a community c should have as large as possible intra-density and as low as possible inter-density. In other words, the difference between the intra-density and the inter-density of a community

$$d_c - \overline{d_c} = \frac{2|E_c^{in}|}{|c|(|c|-1)} - \frac{|E_c^{out}|}{|c||V-c|} \tag{2.6}$$

should be as large as possible. In the formula, d_c denotes the intra-density of community c while $\overline{d_c}$ is the inter-density.

Goldberg et al. [30] stated that overlapping communities should satisfy a minimal set of axioms, (1) connectedness: a community should be a connected subgraph of the network instead of a disconnected one; (2) local optimality: the density $D_s()$ of a community cannot be improved with the addition or removal of a single node. The density function adopted in [30, 31] is

$$D_s(c) = \frac{|E_c^{in}|}{|E_c^{in}| + |E_c^{out}|}. \tag{2.7}$$

In order to handle sparse areas of a graph, Kelley et al. [32] proposed to use the following density function

$$D_s(c) = \frac{|E_c^{in}|}{|E_c^{in}| + |E_c^{out}|} + \lambda d_c, \tag{2.8}$$

where λ is a parameter to fine tune the detected results. A larger value of λ will produce smaller groups, leading to a wide variety of resolutions.

Each community definition above is somehow based on certain community

quality metrics, like *modularity*, the number of *intra-edges*, the number of *inter-edges*, *contraction*, *expansion*, *intra-density*, and *inter-density*. The community can be similarly defined based on other community quality metrics, such as *conductance* [33] and *relative density* [34]. For a community to be "valid", its values of metrics should be as large as possible (or as small as possible), which leads to the idea that optimizing those community quality metrics can lead to the discovery of community structure of networks.

All the above definitions, except the ones in Equations (2.3) and (2.4) which are valid only for disjoint community structure, are applicable both to disjoint and overlapping community structure. It is worth noting that although modularity is suitable to measure overlapping community structure, it was originally intended to disjoint community structure and its value will not be in the range of $[-1, 1]$ for overlapping community structure. Moreover, the definitions of community in Equations (2.3) and (2.4) are global definitions because they are related to the communities neighboring the measured community. In contrast, the other community definitions are local definitions since we can decide whether it is a community based on its own properties regardless how other communities are defined.

2.2 Review of Modularity Related Literature

In this section, we first review the definition of modularity and the corresponding optimization approaches. Then, we discuss the resolution limit problem of modularity maximization. Finally, we overview several community quality measures proposed to resolve this resolution limit problem.

2.2.1 Definition of Modularity

Comparing results of different network community detection algorithms can be challenging, especially when the community structure is not known beforehand. A concept of modularity defined in [5] provides a measure of the quality of a particular partition of a network. Modularity (Q) quantifies the community strength by comparing the fraction of edges within the community with such fraction when random connections between the nodes are made. The justification is that a com-

munity should have more edges between themselves than a random gathering of people. Thus, the Q value close to 0 means that the fraction of edges inside communities is no better than the random case, and the value of 1 means that a network community structure has the highest possible strength.

Formally, for a given community structure C of a network $G = (V, E)$, modularity (Q) can be defined as [5]:

$$Q = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \left(\frac{2\,|E_c^{in}| + |E_c^{out}|}{2\,|E|} \right)^2 \right], \tag{2.9}$$

where C is the set of all the communities, c is a specific community in C, $|E_c^{in}|$ is the number of edges between nodes within community c, $|E_c^{out}|$ is the number of edges from the nodes in community c to the nodes outside c, and $|E|$ is the total number of edges in the network.

Modularity can also be expressed in the following form [8]:

$$Q = \frac{1}{2|E|} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] \delta_{c_i, c_j}, \tag{2.10}$$

where k_i is the degree of node i, A_{ij} is an element of the adjacency matrix between node i and node j, δ_{c_i,c_j} is the Kronecker delta symbol, and c_i is the label of the community to which node i is assigned.

Since larger Q means a stronger community structure, several community detection algorithms, which we will discuss in the next section, are based on modularity optimization.

The modularity measure defined above is suitable only for undirected and unweighted networks. However, this definition can be naturally extended to apply to directed networks as well as to weighted networks. Weighted and directed networks contain more information and are therefore often viewed as more valuable but also as more difficult to analyze than the undirected and unweighted ones.

The revised definition of modularity that works for directed networks is as follows [9]:

$$Q = \frac{1}{|E|} \sum_{ij} \left[A_{ij} - \frac{k_i^{in} k_j^{out}}{|E|} \right] \delta_{c_i, c_j}, \tag{2.11}$$

where k_i^{in} and k_j^{out} are the in- and out- degrees. Or equivalently

$$Q = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \frac{(|E_c^{in}| + |E_{out,c}|)(|E_c^{in}| + |E_{c,out}|)}{|E|^2} \right], \tag{2.12}$$

where $|E_{out,c}|$ is the number of edges from the nodes outside community c to the nodes in c and $|E_{c,out}|$ is the number of edges from the nodes in community c to the nodes outside c. For undirected networks, it is clear that $|E_{out,c}| = |E_{c,out}| = |E_c^{out}|$ and thus the directed modularity is reduced to undirected modularity.

Although many networks can be regarded as binary, i.e. as either having an edge between a pair of nodes or not having it, there are many other networks for which it would be natural to treat edges as having a certain degree of strengths or weights.

The same general techniques that have been developed for unweighted networks are applied to its weighted counterparts in [10] by mapping weighted networks onto multigraphs. For non-negative integer weights, an edge with weight w in a weighted graph corresponds to w parallel edges in a corresponding multigraph. Although negative weights can arise in some applications they are rarely useful in social networks, so for the sake of brevity we will not discuss them here. It turns out that an adjacency matrix of a weighted graph is equivalent to that of a multigraph with unweighted edges. Since the structure of adjacency matrix is independent of the edge weights, it is possible to adjust all the methods developed for unweighted networks to the weighted ones.

It is necessary to point out that the notion of degree of a node should also be extended for the weighted graphs. In this case degree of a node is defined as the sum of weights of all edges incident to this node.

It is shown in [10] that the same definitions of modularity that were given above hold for the weighted networks as well if we treat A_{ij} as the value that represents the weight of the connection. Then, $|E| = \frac{1}{2}\sum_{ij} A_{ij}$ is the sum of the weights of all the edges in the network, $|E_c^{in}|$ is the sum of the weights of the edges between nodes within community c, $|E_c^{out}|$ and $|E_{c,out}|$ is the sum of the weights of the edges from the nodes in community c to the nodes outside c, and $|E_{out,c}|$ is the sum of the

weights of the edges from the nodes outside community c to the nodes in c.

2.2.2 Modularity Optimization Approaches

In the literature, a high value of modularity (Q) indicates a good community structure and the partition corresponding to the maximum value of modularity on a given graph is supposed to have the highest quality, or at least a very good one. Therefore, it is natural to discover communities by maximizing modularity over all possible partitions of a network. However, it is computationally prohibitively expensive to exhaustively search all such partitions for the optimal value of modularity since modularity optimization is known to be NP-hard (Non-deterministic Polynomial-time hard) [35]. However, many heuristic methods were introduced to find high-modularity partitions in a reasonable time. Those approaches include greedy algorithms [7,36–38], spectral methods [8,39–44], extremal optimization [45], simulated annealing [46–49], sampling technique [50], and mathematical programming [51]. In this section, we will review those modularity optimization heuristics.

2.2.2.1 Greedy Algorithms

The first greedy algorithm was proposed by Newman [7]. It is a agglomerative hierarchical clustering method. Initially, every node belongs to its own community, creating altogether $|V|$ communities. Then, at each step, the algorithm repeatedly merges pairs of communities together and chooses the merger for which the resulting modularity is the largest. The change in Q upon joining two communities c_i and c_j is

$$\Delta Q_{c_i,c_j} = 2\left(\frac{|E_{c_i,c_j}|}{2|E|} - \frac{|E_{c_i}||E_{c_j}|}{4|E|^2}\right), \tag{2.13}$$

where $|E_{c_i,c_j}|$ is the number of edges from community c_i to community c_j and $|E_{c_i}| = 2|E^{in}_{c_i}|+|E^{out}_{c_i}|$ is the total degrees of nodes in community c_i. $\Delta Q_{c_i,c_j}$ can be calculated in constant time. The algorithm stops when all the nodes in the network are in a single community after $(|V|-1)$ steps of merging. Then, there are totally $|V|$ partitions, the first one defined by the initial step and each subsequent one resulting from each of the subsequent $(|V|-1)$ merging steps. The partition with the largest value of modularity, approximating the modularity maximum best, is the result of

the algorithm. At each merging step, the algorithm needs to compute the change $\Delta Q_{c_i,c_j}$ of modularity resulting from joining any two currently existing communities c_i and c_j in order to choose the best merger. Since merging two disconnected communities will not increase the value of modularity, the algorithm checks only the merging of connected pairs of communities and the number of such pairs is at most $|E|$ limiting the complexity of this part to $O(|E|)$. However, the rows and columns of adjacency matrix corresponding to the two merged communities must be updated, which takes $O(|V|)$. Since there are $(|V| - 1)$ iterations, the final complexity of the algorithm is $O((|E| + |V|)|V|)$, or $O(|V|^2)$ for sparse networks.

Although Newman's algorithm [7] is much faster than the algorithm of Newman and Girvan [5] whose complexity is $O(|E|^2|V|)$, Clauset et al. [36] pointed out that the update of the adjacency matrix at each step contains a large number of unnecessary operations when the network is sparse and therefore its matrix has a lot of zero entries. They introduced data structures for sparse matrices to perform the updating operation more efficiently. In their algorithm, instead of maintaining the adjacency matrix and computing $\Delta Q_{c_i,c_j}$, they maintained and updated the matrix with entries being $\Delta Q_{c_i,c_j}$ for the pairs of connected communities c_i and c_j. The authors introduced three data structures to represent sparse matrices efficiently: (1) each row of the matrix is stored as a balanced binary tree in order to search and insert elements in $O(log|V|)$ time and also as a max-heap so as to locate the largest element of each row in constant time; (2) another max-heap stores the largest element of each row of the matrix so as to locate the largest $\Delta Q_{c_i,c_j}$ in constant time; (3) a vector is used to save $|E_{c_i}|$ for each community c_i. Then, in each step, the largest $\Delta Q_{c_i,c_j}$ can be found in constant time and the update of the adjacency matrix after merging two communities c_i and c_j takes $O((k_{c_i} + k_{c_j})log|V|)$, where k_{c_i} and k_{c_j} are the numbers of neighboring communities of communities c_i and c_j, respectively. Thus, the total running time is at most $O(log|V|)$ times the sum of the degrees of nodes in the communities along the dendrogram created by merging steps. This sum is in the worst case the depth of the dendrogram times the sum of the degrees of nodes in the network. Suppose the dendrogram has depth d, then the running time is $O(d|E|log|V|)$, or $O(|V|log^2|V|)$ when the network is sparse and the

dendrogram is almost balanced ($d \sim log|V|$).

However, Wakita and Tsurumi [37] observed that the greedy algorithm proposed by Clauset et al. is not scalable to networks with sizes larger than $500,000$ nodes. They found that the computational inefficiency arises from merging communities in an unbalanced manner, which yields very unbalanced dendrograms. In such cases, the relation $d \sim log|V|$ does not hold any more, causing the algorithm to run at its worst-case complexity. To balance the merging of communities, the authors introduced three types of *consolidation ratios* to measure the balance of the community pairs and used it with modularity to perform the joining process of communities without bias. This modification enables the algorithm to scale to networks with sizes up to $10,000,000$. It also approximates the modularity maximum better than the original algorithm.

Another type of greedy modularity optimization algorithm different from those above was proposed by Blondel et al., and it is usually referred to as Louvain [38]. It is divided into two phases that are repeated iteratively. Initially, every node belongs to the community of itself, so there are $|V|$ communities. In this first phase, every node, in a certain order, is considered for merging into its neighboring communities and the merger with the largest positive gain is selected. If all possible gains associated with the merging of this node are negative, then it stays in its original community. This merging procedure repeats iteratively and stops when no increase of Q can be achieved.

After the first phase, Louvain reaches a local maximum of Q. Then, the second phase of Louvain builds a community network based on the communities discovered in the first phase. The nodes in the new network are the communities from the first phase and there is a edge between two new nodes if there are edges between nodes in the corresponding two communities. The weights of those edges are the sum of the weights of the edges between nodes in the corresponding two communities. The edges between nodes of the same community of the first phase result in a self-loop for this community node in the new network. After the community network is generated, the algorithm applies the first phase again on this new network. The two phases repeat iteratively and stop when there is no more change and consequently

a maximum modularity is obtained. The number of iterations of this algorithm is usually very small and most of computational time is spent in the first iteration. Thus, the complexity of the algorithm grows like $O(|E|)$. Consequently, it is scalable to large networks with the number of nodes up to a billion. However, the results of Louvain are impacted by the order in which the nodes in the first phase are considered for merging [52].

2.2.2.2 Spectral Methods

There are two categories of spectral algorithms for maximizing modularity: one is based on the modularity matrix [8,39,40]; the other is based on the Laplacian matrix of a network [41–43].

***A*. Modularity optimization using the eigenvalues and eigenvectors of the modularity matrix [8, 39, 40].**

Modularity (Q) can be expressed as [8]

$$\begin{aligned} Q &= \frac{1}{4|E|}\sum_{ij}\left(A_{ij}-\frac{k_i k_j}{2|E|}\right)(s_i s_j+1) \\ &= \frac{1}{4|E|}\sum_{ij}\left(A_{ij}-\frac{k_i k_j}{2|E|}\right)s_i s_j \\ &= \frac{1}{4|E|}\boldsymbol{s}^T\boldsymbol{B}\boldsymbol{s}, \end{aligned} \tag{2.14}$$

where A_{ij} are the elements of adjacency matrix $\boldsymbol{A}$ and $\boldsymbol{s}$ is the column vector representing any division of the network into two groups. Its elements are defined as $s_i = +1$ if node i belongs to the first group and $s_i = -1$ if it belongs to the second group. $\boldsymbol{B}$ is the modularity matrix with elements

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2|E|}. \tag{2.15}$$

Representing $\boldsymbol{s}$ as a linear combination of the normalized eigenvectors $\boldsymbol{u_i}$ of $\boldsymbol{B}$: $\boldsymbol{s} = \sum_{i=1}^{|V|} a_i \boldsymbol{u_i}$ with $a_i = \boldsymbol{u_i}^T \cdot \boldsymbol{s}$, and then plugging the result into Equation (2.14)

yield

$$\begin{aligned} Q &= \frac{1}{4|E|} \sum_i a_i \boldsymbol{u_i}^T \boldsymbol{B} \sum_j a_j \boldsymbol{u_j} \\ &= \frac{1}{4|E|} \sum_i a_i^2 \beta_i, \end{aligned} \tag{2.16}$$

where β_i is the eigenvalue of $\boldsymbol{B}$ corresponding to eigenvector $\boldsymbol{u_i}$. To maximize Q above, Newman [8] proposed a spectral approach to choose $\boldsymbol{s}$ proportional to the leading eigenvector $\boldsymbol{u_1}$ corresponding to the largest (most positive) eigenvalue β_1. The choice assumes that the eigenvalues are labeled in decreasing order $\beta_1 \geq \beta_2 \geq ... \geq \beta_{|V|}$. Nodes are then divided into two communities according to the signs of the elements in $\boldsymbol{s}$ with nodes corresponding to positive elements in $\boldsymbol{s}$ assigned to one group and all remaining nodes to another. Since the row and column sums of $\boldsymbol{B}$ are zero, it always has an eigenvector $(1, 1, 1, ...)$ with eigenvalue zero. Therefore, if it has no positive eigenvalue, then the leading eigenvector is $(1, 1, 1, ...)$, which means that the network is indivisible. Moreover, Newman [8] proposed to divide network into more than two communities by repeatedly dividing each of the communities obtained so far into two until the additional contribution ΔQ to the modularity made by the subdivision of a community c

$$\begin{aligned} \Delta Q &= \frac{1}{2|E|} \left[\frac{1}{2} \sum_{i,j \in c} B_{ij}(s_i s_j + 1) - \sum_{i,j \in c} B_{ij} \right] \\ &= \frac{1}{4|E|} \boldsymbol{s}^T \boldsymbol{B^{(c)}} \boldsymbol{s} \end{aligned} \tag{2.17}$$

is equal to or less than 0. $\boldsymbol{B^{(c)}}$ in the formula above is the generalized modularity matrix. Its elements, indexed by the labels i and j of nodes within community c, are

$$B_{ij}^{(c)} = B_{ij} - \delta_{ij} \sum_{k \in c} B_{ik}. \tag{2.18}$$

Then, the same spectral method can be applied to $\boldsymbol{B^{(c)}}$ to maximize ΔQ. The recursive subdivision process stops when $\Delta Q \leq 0$, which means that there is no positive eigenvalue of the matrix $\boldsymbol{B^{(c)}}$. The overall complexity of this algorithm is

$O((|E|+|V|)|V|)$.

However, the spectral algorithm described above has two drawbacks. First, it divides a network into more than two communities by repeated division instead of getting all the communities directly in a single step. Second, it only uses the leading eigenvector of the modularity matrix and ignores all the others, losing all the useful information contained in those eigenvectors. Newman later proposed to divide a network into a set of communities C with $|C| \geq 2$ directly using multiple leading eigenvectors [39]. Let $\boldsymbol{S} = (\boldsymbol{s_c})$ be an $|V| \times |C|$ "community-assignment" matrix with one column for each community c defined as

$$S_{i,c} = \begin{cases} 1 & \text{if node } i \text{ belongs to community } c, \\ 0 & \text{otherwise.} \end{cases} \tag{2.19}$$

then the modularity (Q) for this direct division of the network is given by

$$Q = \frac{1}{2|E|} \sum_{i,j=1}^{|V|} \sum_{c \in C} B_{ij} S_{i,c} S_{j,c} = \frac{1}{2|E|} \mathrm{Tr}(\boldsymbol{S}^T \boldsymbol{B} \boldsymbol{S}), \tag{2.20}$$

where $\mathrm{Tr}(\boldsymbol{S}^T \boldsymbol{B} \boldsymbol{S})$ is the trace of matrix $\boldsymbol{S}^T \boldsymbol{B} \boldsymbol{S}$. Defining $\boldsymbol{B} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{U}^T$, where $\boldsymbol{U} = (\boldsymbol{u_1}, \boldsymbol{u_2}, ...)$ is the matrix of eigenvectors of $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$ is the diagonal matrix of eigenvalues $\Sigma_{ii} = \beta_i$, yields

$$Q = \frac{1}{2|E|} \sum_{i=1}^{|V|} \sum_{c \in C} \beta_i (\boldsymbol{u_i}^T \boldsymbol{s_c})^2. \tag{2.21}$$

Then, obtaining $|C|$ communities is equivalent to selecting $|C| - 1$ independent, mutually orthogonal columns $\boldsymbol{s_c}$. Moreover, Q would be maximized by choosing the columns $\boldsymbol{s_c}$ proportional to the leading eigenvectors of $\boldsymbol{B}$. However, only the eigenvectors corresponding to the positive eigenvalues will contribute positively to the modularity. Thus, the number of positive eigenvalues, plus 1, is the upper bound of $|C|$. More general modularity maximization is to keep the leading p ($1 \leq p \leq |V|$)

eigenvectors. Q can be rewritten as

$$\begin{aligned} Q &= \frac{1}{2|E|}\left(|V|\alpha + \mathrm{Tr}[\boldsymbol{S}^T\boldsymbol{U}(\boldsymbol{\Sigma}-\alpha\boldsymbol{I})\boldsymbol{U}^T\boldsymbol{S}]\right) \\ &= \frac{1}{2|E|}\left(|V|\alpha + \sum_{j=1}^{|V|}\sum_{c\in C}(\beta_j-\alpha)\left[\sum_{i=1}^{|V|}U_{ij}S_{i,c}\right]^2\right), \end{aligned} \tag{2.22}$$

where α ($\alpha \leq \beta_p$) is a constant related to the approximation for Q obtained by only adopting the first p leading eigenvectors. By selecting $|V|$ node vectors $\boldsymbol{r_i}$ of dimension p whose jth component is

$$[\boldsymbol{r_i}]_j = \sqrt{\beta_j - \alpha}U_{ij}, \tag{2.23}$$

modularity can be approximated as

$$Q \simeq \widetilde{Q} = \frac{1}{2|E|}\left(|V|\alpha + \sum_{c\in C}|\boldsymbol{R_c}|^2\right), \tag{2.24}$$

where $\boldsymbol{R_c}$, $c \in C$, are the community vectors

$$\boldsymbol{R_c} = \sum_{i\in c}\boldsymbol{r_i}. \tag{2.25}$$

Thus, the community detection problem is equivalent to choosing such a division of nodes into $|C|$ groups that maximizes the magnitudes of the community vectors $\boldsymbol{R_c}$ while requiring that $\boldsymbol{R_c}\cdot\boldsymbol{r_i} > 0$ if node i is assigned to community c. Problems of this type are called *vector partitioning* problems.

Although [39] explored using multiple leading eigenvectors of the modularity matrix, it did not pursue it in detail beyond a two-eigenvector approach for bipartitioning [8, 39]. Richardson et al. [40] provided a extension of these recursive bipartitioning methods by considering the best two-way or three-way division at each recursive step to more thoroughly explore the promising partitions. To reduce the number of partitions considered for the eigenvector-pair tripartitioning, the authors adopted a divide-and-conquer method and as a result yielded an efficient approach whose computational complexity is competitive with the two-eigenvector

bipartitioning method.

***B.* Modularity optimization using the eigenvalues and eigenvectors of the Laplacian matrix [41–43].**

Given a partition C (a set of communities) and the corresponding "community-assignment" matrix $\boldsymbol{S} = (\boldsymbol{s_c})$, White and Smyth [41] rewrote modularity (Q) as follows:

$$Q \propto \mathrm{Tr}(\boldsymbol{S}^T(\boldsymbol{W} - \widetilde{\boldsymbol{D}})\boldsymbol{S}) = -\mathrm{Tr}(\boldsymbol{S}^T\boldsymbol{L_Q}\boldsymbol{S}), \tag{2.26}$$

where $\boldsymbol{W} = 2|E|\boldsymbol{A}$ and the elements of $\widetilde{\boldsymbol{D}}$ are $\widetilde{D}_{ij} = k_i k_j$. The matrix $\boldsymbol{L_Q} = \widetilde{\boldsymbol{D}} - \boldsymbol{W}$ is called the "Q-Laplacian". Finding the "community-assignment" matrix $\boldsymbol{S}$ that maximizes Q above is NP-complete, but a good approximation can be obtained by relaxing the discreteness constraints of the elements of $\boldsymbol{S}$ and allowing them to assume real values. Then, Q becomes a continuous function of $\boldsymbol{S}$ and its extremes can be found by equating its first derivative with respect to $\boldsymbol{S}$ to zero. This leads to the eigenvalue equation:

$$\boldsymbol{L_Q}\boldsymbol{S} = \boldsymbol{S}\boldsymbol{\Lambda}, \tag{2.27}$$

where $\boldsymbol{\Lambda}$ is the diagonal matrix of Lagrangian multipliers. Thus, the modularity optimization problem is transformed into the standard spectral graph partitioning problem. When the network is not too small, $\boldsymbol{L_Q}$ can be approximated well, up to constant factors, by the transition matrix $\widetilde{\boldsymbol{W}} = \boldsymbol{D}^{-1}\boldsymbol{A}$ obtained by normalizing $\boldsymbol{A}$ so that all rows sum to one. $\boldsymbol{D}$ here is the diagonal degree matrix of $\boldsymbol{A}$. It can be shown that the eigenvalues and eigenvectors of $\widetilde{\boldsymbol{W}}$ are precisely $1 - \lambda$ and $\boldsymbol{\mu}$, where λ and $\boldsymbol{\mu}$ are the solutions to the generalized eigenvalue problem $\boldsymbol{L}\boldsymbol{\mu} = \lambda\boldsymbol{D}\boldsymbol{\mu}$ where $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ is the Laplacian matrix. Thus, the underlying spectral algorithm here is equivalent to the standard spectral graph partitioning problem which uses the eigenvalues and eigenvectors of the Laplacian matrix.

Based on the above analysis, White and Smyth proposed two clustering algorithms, named "Algorithm Spectral-1" and "Algorithm Spectral-2", to search for a partition C with size up to K predefined by an input parameter. Both algorithms take the eigenvector matrix $\boldsymbol{U_K} = (\boldsymbol{u_1}, \boldsymbol{u_2}, ..., \boldsymbol{u_{K-1}})$ with the leading $K - 1$ eigenvectors (excluding the trivial all-ones eigenvector) of the transition matrix $\widetilde{\boldsymbol{W}}$ as

input. Those $K-1$ eigenvectors can be efficiently computed with the Implicitly Restarted Lanczos Method (IRLM) [53]. "Algorithm Spectral-1" uses the first $k-1$ ($2 \leq k \leq K$) columns of $\boldsymbol{U_K}$, denoted as $\boldsymbol{U_{k-1}}$, and clusters the row vectors of $\boldsymbol{U_{k-1}}$ using k-means to find a k-way partition, denoted as C_k. Then, the C_{k^*} with size k^* that achieves the largest value of Q is the final community structure.

"Algorithm Spectral-2" starts with a single community ($k = 1$) and recursively splits each community c into two smaller ones if the subdivision produces a higher value of Q. The split is done by running k-means with two clusters on the matrix $\boldsymbol{U_{k,c}}$ formed from $\boldsymbol{U_k}$ by keeping only rows corresponding to nodes in c. The recursive procedure stops when no more splits are possible or when $k = K$ communities have been found and then the final community structure with the highest value of Q is the detection result.

However, the two algorithms described above, especially "Algorithm Spectral-1", scale poorly to large networks because of running k-means partitioning up to K times. Both approaches have a worst-case complexity $O(K^2|V| + K|E|)$. In order to speed up the calculation while retaining effectiveness in approximating the maximum of Q, Ruan and Zhang [42] proposed the *Kcut* algorithm which recursively partitions the network to optimize Q. At each recursive step, *Kcut* adopts a k-way partition ($k = 2, 3, ..., l$) to the subnetwork induced by the nodes and edges in each community using "Algorithm Spectral-1" of White and Smyth [41]. Then, it selects the k that achieves the highest Q. Empirically, *Kcut* with l as small as 3 or 4 can significantly improve Q over the standard bi-partitioning method and it also reduces the computational cost to $O((|V| + |E|)log|C|)$ for a final partition with $|C|$ communities.

Ruan and Zhang later [43] proposed $QCUT$ algorithm that combines *Kcut* and local search to optimize Q. $QCUT$ stands for modularity (Q) cut (partitioning). The $QCUT$ algorithm consists of two alternating stages: partitioning and refinement. In the partitioning stage, *Kcut* is used to recursively partition the network until Q cannot be further improved. In the refinement stage, a local search strategy repeatedly considers two operations. The first one is migration that moves a node from its current community to another one and the second one is the merge

of two communities into one. Both are applied to improve Q as much as possible. The partitioning stage and refinement stage are alternating until Q cannot be increased further. In order to solve the resolution limit problem of modularity, the authors proposed $HQCUT$ (hierarchical $QCUT$) which recursively applies $QCUT$ to divide the subnetwork, generated with the nodes and edges in each community, into subcommunities. Further, to avoid overpartitioning, they use a statistical test to determine whether a community indeed has intrinsic subcommunities.

***C.* Equivalence of two categories of spectral algorithms for maximizing modularity [44].**

Newman [44] showed that with hyperellipsoid relaxation, the spectral modularity maximization method using the eigenvalues and eigenvectors of the modularity matrix can be formulated as the spectral algorithm that relies on the eigenvalues and eigenvectors of Laplacian matrix. This formulation indicates that the above two kinds of modularity optimization approaches are equivalent. Starting with Equation (2.14) for the division of a network into two groups, first the discreteness of s_i is relaxed onto a hyperellipsoid with the constraint

$$\sum_i k_i s_i^2 = 2|E|. \tag{2.28}$$

Then, the relaxed modularity maximization problem can be easily solved by setting the first derivative of Equation (2.14) with respect to s_i to zero. This leads to

$$\sum_j B_{ij} s_j = \lambda k_i s_i, \tag{2.29}$$

or in matrix notation

$$\boldsymbol{B}\boldsymbol{s} = \lambda \boldsymbol{D}\boldsymbol{s}, \tag{2.30}$$

where λ is the eigenvalue. Plugging Equation (2.29) into Equation (2.14) yields

$$Q = \frac{1}{4|E|} \sum_{ij} B_{ij} s_i s_j = \frac{\lambda}{4|E|} \sum_i k_i s_i^2 = \frac{\lambda}{2}. \tag{2.31}$$

Therefore, to achieve the highest value of Q, one should chose λ to be the largest

(most positive) eigenvalue of Equation (2.30). Using Equation (2.15), Equation (2.29) can be rewritten as

$$\sum_j A_{ij}s_j = k_i(\lambda s_i + \frac{1}{2|E|}\sum_j k_j s_j), \tag{2.32}$$

or in matrix notion as

$$\boldsymbol{A}\boldsymbol{s} = \boldsymbol{D}\left(\lambda\boldsymbol{s} + \frac{\boldsymbol{k}^T\boldsymbol{s}}{2|E|}\mathbf{1}\right), \tag{2.33}$$

where $\boldsymbol{k}$ is the vector with element k_i and $\mathbf{1} = (1, 1, 1, ...)$. Then, multiplying the above equation by $\mathbf{1}^T$ results in $\lambda\boldsymbol{k}^T\boldsymbol{s} = 0$. If there is a nontrivial eigenvalue $\lambda > 0$, then the above equation simplifies to

$$\boldsymbol{A}\boldsymbol{s} = \lambda\boldsymbol{D}\boldsymbol{s}. \tag{2.34}$$

Again, λ should be the most positive eigenvalue. However, the eigenvector corresponding to this eigenvalue is the uniform vector $\mathbf{1}$ which fails to satisfy $\boldsymbol{k}^T\boldsymbol{s} = 0$. Thus, in this case, one can do the best by choosing λ to be the second largest eigenvalue and having $\boldsymbol{s}$ proportional to the corresponding eigenvector. In fact, this eigenvector is precisely equal to the leading eigenvector of Equation (2.30). Then, after defining a rescaled vector $\boldsymbol{u} = \boldsymbol{D}^{1/2}\boldsymbol{s}$ and plugging it into Equation (2.34), we get

$$(\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2})\boldsymbol{u} = \lambda\boldsymbol{u}. \tag{2.35}$$

The matrix $\boldsymbol{L} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ is called the normalized Laplacian matrix. (The normalized Laplacian is sometimes defined as $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$, but those two differ only by a trivial transformation of their eigenvalues and eigenvectors.)

2.2.2.3 Extremal Optimization

Duch and Arenas [45] proposed a modularity optimization algorithm based on the Extremal Optimization (EO) [54]. EO optimizes a global variable by improving extremal local variables. Here, the global variable is modularity (Q). The contribution of an individual node i to Q of the whole network with a certain community structure is given by

$$q_i = k_{i,c_i} - k_i\frac{|E_{c_i}|}{2|E|}, \tag{2.36}$$

where k_{i,c_i} is the number of edges that connect node i to the nodes in its own community c_i. Notice that $Q = \frac{1}{2|E|}\sum_i q_i$ and q_i can be normalized into the interval $[-1, 1]$ by diving it by k_i

$$\lambda_i = \frac{q_i}{k_i} = \frac{k_{i,c_i}}{k_i} - \frac{|E_{c_i}|}{2|E|}, \tag{2.37}$$

where λ_i, called fitness, is the relative contribution of node i to Q. Then, the fitness of each node is adopted as the local variable.

The algorithm starts by randomly splitting the network into two partitions of equal number of nodes, where communities are the connected components in each partition. Then, at each iteration, it moves the node with the lowest fitness from its own community to another community. The shift changes the community structure, so the fitness of many other nodes needs to be recomputed. The process repeats until it cannot increase Q. After that, it generates sub-community networks by deleting the inter-community edges and proceeds recursively on each sub-community network until Q cannot be improved. Although the procedure is deterministic when given the initialization, its final result in fact depends on the initialization and it is likely to get trapped in local maxima. Thus, a probabilistic selection called τ-EO [54] in which nodes are ranked according to their fitness and a node of rank r is selected with the probability $P(r) \propto r^{-\tau}$ is used to improve the result. The computational complexity of this algorithm is $O(|V|^2 log^2 |V|)$.

2.2.2.4 Simulated Annealing

Simulated annealing (SA) [55] is a probabilistic procedure for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. This technique was adopted in [46–49] to maximize modularity (Q). The initial point for all those approaches can be arbitrary partitioning of nodes into communities, even including $|V|$ communities in which each node belongs to its own community. At each iteration, a node i and a community c are chosen randomly. This community could be a currently existing community or an empty community introduced to increase the number of communities. Then, node i is moved from its original community to this new community c, which would change Q by ΔQ. If ΔQ is greater than zero, this update is accepted, otherwise

it is accepted with probability $e^{\beta \Delta Q}$ where β in [46–48] represents the inverse of temperature T and β in [49] is the reciprocal of pseudo temperature τ. In addition in [49], there is one more condition for the move of a node when c is not empty, shifting node i to c is considered only if there are some edges between node i and the nodes in c. To improve the performance and to avoid getting trapped in local minima, collective movements which involve moving multiple nodes at a time [48, 49], merging two communities [46–48], and splitting a community [46–48] are employed. Splits can be carried out in a number of different schemes. The best performance is achieved by treating a community as an isolated subnetwork and partitioning it into two and then performing a nested SA on these partitions [46, 47]. Those methods stop when no new update is accepted within a fixed number of iterations.

2.2.2.5 Sampling Techniques

Sales-Pardo et al. [50] proposed a "box-clustering" method to extract the hierarchical organization of networks. This approach consists of two steps: (1) estimating the similarity, called "node affinity", between nodes and forming the node affinity matrix; (2) deriving hierarchical community structure from the affinity matrix. The affinity between two nodes is the probability that they are classified into the same community in the local maxima partitions of modularity. The set of local maxima partitions, called P_{max}, includes those partitions for which neither the moving of a node from its original community to another, nor the merging of two communities will increase the value of modularity. The sample P_{max} is found by performing the simulated annealing based modularity optimization algorithm of Guimerá and Amaral [46, 47]. More specifically, the algorithm first randomly divides the nodes into communities and then performs the hill-climbing search until a sample with local maximum of modularity is reached. Then, the affinity matrix is updated based on the obtained sample.

The sample generation procedure is repeated until the affinity matrix has converged to its asymptotic value. Empirically, the total number of samples needed is proportional to the size of the network. Before proceeding to the second step, the algorithm assesses whether the network has a significant community structure

or not. It is done by computing the z-score of the average modularity of the partitions in P_{max} with respect to the average modularity of the partitions with the local modularity maxima of the equivalent ensemble of null model networks. The equivalent null model is obtained by randomly rewiring the edges of the original network while retaining the degree sequence. Large z-score indicates that the network has a meaningful internal community structure. If the network indeed has a significant community structure, the algorithm advances to the second step to group nodes with large affinity close to each other. The goal is to bring the form of the affinity matrix as close as possible to block-diagonal structure by minimizing the cost function representing the average distance of matrix elements to the diagonal. Then, the communities corresponds to the "best" set of boxes obtained by least-squares fitting of the block-diagonal structure to the affinity matrix. The procedure described above can be recursively performed to subnetworks induced by communities to identify the low level structure of each community until no subnetwork is found to have significant intrinsic structure.

2.2.2.6 Mathematical Programming

Agarwal and Kempe [51] formulated the modularity maximization problem as a linear program and vector program which have the advantage of providing a posteriori performance guarantees. First, modularity maximization can be transformed into the integer program

$$\begin{aligned} &\text{Maximize } \frac{1}{2|E|}\sum_{ij} B_{ij}(1-x_{ij}) \\ &\text{subject to } x_{ik} \leq x_{ij} + x_{jk} \text{ for all } i,j,k \\ &\qquad x_{ij} \in \{0,1\} \text{ for all } i,j, \end{aligned} \tag{2.38}$$

where $\boldsymbol{B}$ is the modularity matrix and the objective function is linear in the variable x_{ij}. When $x_{ij} = 0$, i and j belong to the same community and $x_{ij} = 1$ indicates that they are in different communities. The restriction $x_{ik} \leq x_{ij} + x_{jk}$ requires that i and k are in the same community if and only if i, j, and k are in the same community. Solving the above integer program is NP-hard, but relaxing the last

constraint that x_{ij} is a integer from $\{0, 1\}$ to allow x_{ij} be a real number in the interval $[0, 1]$ reduces the integer program to a linear program which can be solved in polynomial time [56]. However, the solution does not correspond to a partition when any of x_{ij} is fractional. To get the communities from x_{ij}, a rounding step is needed. The value of x_{ij} is treated as the distance between i and j and these distances are used repeatedly to form communities of "nearby" nodes. Moreover, optimizing modularity by dividing a network into two communities can be considered as a strict quadratic program

$$\begin{aligned} &\text{Maximize } \frac{1}{4|E|}\sum_{ij} B_{ij}(1 + s_i s_j) \\ &\text{subject to } s_i^2 = 1 \text{ for all } i, \end{aligned} \tag{2.39}$$

where the objective function is the same as Equation (2.14) defined by Newman [8]. Note that the constraint $s_i^2 = 1$ ensures that $s_i = \pm 1$ which implies that node i belongs either to the first or the second community. Quadratic programming is NP-complete, but it could be relaxed to a vector program by replacing each variable s_i with $|V|$-dimensional vector $\boldsymbol{s}$ and replacing the scalar product with the inner vector product. The solution to vector program is one location per node on the surface of a $|V|$-dimensional hypersphere. To obtain a bipartition from these node locations, a rounding step is needed which chooses any random $(|V|-1)$-dimensional hyperplane passing through the origin and uses this hyperplane to cut the hypersphere into two halves and as a result separate the node vectors into two parts. Multiple random hyperplanes can be chosen and the one that gets the community structure with the highest modularity provides a solution. The same vector program is then recursively applied to subnetworks generated with nodes and edges in discovered communities to get hierarchical communities until Q cannot be increased. Following the linear program and vector program, Agarwal and Kempe also adopted a post-processing step similar to the local search strategy proposed by Newman [8] to further improve the results.

2.2.3 Resolution Limit Problem

Since its inception, the modularity has been used extensively as the measure of the quality of community structure produced by community detection algorithms. In fact, if we adopt modularity as a quality measure of network partitions, the task of discovering communities is essentially turned into the task of finding the community structure with an optimal value of modularity.

However as the properties of modularity were studied, it was discovered that in some cases it fails to detect small communities. There is a certain threshold [11], such that a community of the size below it will not be detected even if it is a complete subgraph connected to the rest of the graph with a single edge. This property of modularity has become known as the *resolution limit*.

Although the resolution limit prevents detection of small communities, the actual value of the threshold depends on the total number of edges in the network and on the degree of interconnectedness between communities. In fact, the resolution limit can reach the values comparable to the size of the entire network causing formation of a few giant communities (or even a single community) and failing to detect smaller communities within them. It makes interpreting the results of community detection very difficult because it is impossible to tell beforehand whether a community is well-formed or if it can be further split into subcommunities.

Considering modularity as a function of the total number of edges, $|E|$, and the number of communities, m, makes it possible to find the values of m and $|E|$ which maximize this function. It turns out that setting $m = \sqrt{|E|}$ yields the absolute maximal value of modularity. Consequently, modularity has a resolution limit of order $\sqrt{|E|}$ which bounds the number and also the size of communities [11]. In fact, if for a certain community the number of edges inside it is smaller than $\sqrt{\frac{|E|}{2}}$, such community cannot be resolved through the modularity optimization. It is also possible for modularity optimization to fail to detect communities of larger size if they have more edges in common with the rest of the network. Therefore, by finding the optimal value of the modularity we are generally not obtaining the best possible structure of communities.

The above arguments of resolution limit can also be applied to weighted net-

works. In this case, $|E|$ is the sum of the weights of all the edges in the network, $|E_c^{in}|$ is the sum of the weights of the edges between nodes within community c, and $|E_c^{out}|$ is the sum of the weights of the edges from the nodes in community c to the nodes outside c.

By introducing an additional parameter, ϵ, which represents the weight of inter-community edges, Berry et al. showed in [57] that the number of communities in the optimal solution is

$$m = \sqrt{\frac{|E|}{\epsilon}}. \tag{2.40}$$

Correspondingly, any community for which its size

$$|c| < \sqrt{\frac{|E|\epsilon}{2}} - \epsilon \tag{2.41}$$

may not be resolved.

Introduction of ϵ brings some interesting opportunities. If we can make ϵ arbitrarily small, then we can expect maximum weighted modularity to produce any desired number of communities. In other words, given a proper weighting, a much better modularity resolution can be achieved than without weighting. However, in practice, finding a way to set edge weights to achieve small values of ϵ can be challenging. An algorithm for lowering ϵ proposed by Berry et al. requires $O(m|V|\log|V|)$ time.

2.2.4 Resolving the Resolution Limit Problem

There have been extensive studies done on how to mitigate the consequences of the resolution limit of modularity. The main approaches followed are described below.

Localized modularity measure (LQ) [20] is based on the observation that the resolution limit problem is caused by modularity being a global measure since it assumes that edges between any pair of nodes are equally likely, including connectivity between the communities. However, in many networks, the majority of communities have edges to only a few other communities, i.e. exhibit a local community connectivity.

Thus, a local version of the modularity measure for an undirected network is defined as:

$$LQ = \sum_{c \in C} \left[\frac{|E_c^{in}|}{\left|E_c^{neighb}\right|} - \left(\frac{2\,|E_c^{in}| + |E_c^{out}|}{2\left|E_c^{neighb}\right|} \right)^2 \right], \tag{2.42}$$

where $\left|E_c^{neighb}\right|$ is the total number of edges of the subnetwork consisting of community c and the neighboring communities of c.

Unlike the traditional modularity (Q), the local version of modularity (LQ) is not bounded above by 1. The more locally connected communities a network has, the bigger its LQ can grow. In a network where all communities are connected to each other, LQ yields the same value as Q. LQ considers individual communities and their neighbors, and therefore provides a measure of community quality that is not dependent on other parts of the network. The local connectivity approach can be applied not only to the nearest neighboring communities, but also to the second or higher layers of neighbors as well.

Arenas et al. proposed a multiple resolution method [58] which is based on the idea that it might be possible to look at the detected community structure at different scales. From this perspective, the modularity resolution limit is not a problem but a feature. It allows choosing a desired resolution level to achieve the required granularity of the output community structure using the original definition of modularity.

The multiple resolution method is based on the definition of modularity given by Equation (2.9). The modularity resolution limit depends on the total weight $2\,|E|$. By varying the total weight, it is possible to control the resolution limit, effectively performing community detection at different granularity levels. Changing the sum of weights of edges adjacent to every node by some value r results in rescaling topology by a factor of r. Since the resolution limit is proportional to $\sqrt{r}$, the growth of the resolution limit is slower than that of r. Consequently, it would be possible to achieve a scale at which all required communities would be visible to the modularity optimization problem.

Caution should be exercised when altering the weights of edges in the network to avoid changing its topological characteristics. To ensure this, a rescaled adjacency

matrix can be defined as:

$$\boldsymbol{A_r} = \boldsymbol{A} + r\boldsymbol{I}, \tag{2.43}$$

where $\boldsymbol{A}$ is the adjacency matrix and $\boldsymbol{I}$ is the identity matrix. Since the original edge weights are not altered, $\boldsymbol{A_r}$ preserves all common features of the network: distribution of the sum of the weights of the edges incident to each node, weighted clustering coefficient, eigenvectors, etc. Essentially, introducing r results in a self-loop of weight r being added to every node in the network.

Optimizing the modularity for the rescaled topology A_r is performed by using the modularity at scale r as the new quality function:

$$Q_r = \sum_{c \in C} \left[\frac{2\,|E_c^{in}| + r\,|c|}{2\,|E| + r\,|V|} - \left(\frac{|E_c| + r\,|c|}{2\,|E| + r\,|V|} \right)^2 \right], \tag{2.44}$$

where $|c|$ is the number of nodes in community c and $|E_c| = 2|E_c^{in}| + |E_c^{out}|$. It yields larger communities for smaller values of r and smaller communities for larger values of r. By performing modularity optimization for different values of r, it is possible to analyze the community structure at different scales.

Parameter r can also be thought of as representing resistance of a node to become part of a community. If r is positive, we can obtain a community structure that is more granular than what is possible to achieve with the original definition of modularity (Q) which corresponds to r being zero. Making r negative zooms out of the network and provides a view of super communities. Typically, it is not clear how to choose the correct value for this parameter.

Further studies of the multiple resolution approach revealed that it suffers from two major issues outlined in [59]. First, when the value of the resolution parameter r is low, it tends to group together small communities. Second, when the resolution parameter r is high, it splits large communities. These trends are opposite for networks with a large variation of community sizes. Hence, it is impossible to select a value of the resolution parameter such that neither smaller nor larger communities are adversely affected by the resolution limit. A network can be tested for susceptibility to the resolution problem by examining its clustering coefficient, i.e. a degree to which nodes tend to form communities. If the clustering coefficient has

sharp changes, it indicates that communities of substantially different scales exist in this network. The result is that when the value of r is sufficiently large, bigger communities get broken up before smaller communities are found. This applies also to other multiple resolution methods and seems to be a general problem of the methods that are trying to optimize some global measures.

The hierarchical multiresolution method proposed by Granell et al. in [60] overcomes the limitations of the multiple resolution method on networks with very different scales of communities. It achieves that by introducing a new hierarchical multiresolution scheme that works even in cases of community detection near the modularity resolution limit. The main idea underlying this method is based on performing multiple resolution community detection on essential parts of the network, thus analyzing each part independently.

The method operates iteratively by first placing all nodes in a singe community. Then, it finds the minimum value of the resistance parameter r which produces a community structure with the optimal value of modularity. Finally, it runs the same algorithm on each community that was found. The method terminates when no more split of communities is necessary, which usually takes just a few steps.

In the study [28] by Li, Zhang et al., a new quantitative measure for the quality of community structure is introduced. It offers several improvements over the modularity (Q), including elimination of the resolution limit and ability to detect the number of communities. The new measure called modularity density (D) is based on the average degree of the community structure. It is given by:

$$D = \sum_{c \in C} \frac{2\,|E_c^{in}| - |E_c^{out}|}{|c|}. \tag{2.45}$$

The quality of the discovered communities is then described by the value of the modularity density (D). The larger the value of D is, the stronger the community structure is.

The modularity density (D) does not divide a clique into two parts, and it can resolve most modular networks correctly. It can also detect communities of different sizes. This second property can be used to quantitatively determine the number of

communities, since the maximum D value is achieved when the network is supposed to be correctly partitioned. Although as mentioned in [28] finding an optimal value of modularity density (D) is NP-hard, it is equivalent to an objective function of the kernel k means clustering problem for which efficient computational algorithms are known.

Traag et al. in [61] introduced a rigorous definition of the *resolution-limit-free* method for which considering any induced subgraph of the original graph does not cause the detected community structure to change. In other words, if there is an optimal partition of a network (with respect to some objective function), and for each subgraph induced by the partitioning it is also optimal, then such objective function is called resolution-limit-free. An objective function is called *additive* for a certain partition if it is equal to the sum of the values of this objective function for each of the subgraphs induced by the partitioning.

Based on these two definitions it is proved that if an objective function is additive and there are two optimal partitions, then any combination of these partitions is also optimal. In case of a complete graph, if an objective function is resolution-limit-free, then an optimal partition either contains all the nodes (i.e. there is only one community which includes all nodes) or consists of communities of size 1 (i.e. each node forms a community of its own). A more general statement for arbitrary objective functions is also true: if an objective function has local weights (i.e. weights that do not change when considering subgraphs) then it is resolution-limit-free. Although the converse is not true, there is only a relatively small number of special cases when methods with non-local weights are resolution-limit-free.

The authors then analyze *resolution-limit-free* within the framework of the first principle Potts model [62]:

$$\mathcal{H} = -\sum_{ij} \left(a_{ij} A_{ij} - b_{ij} \left(1 - A_{ij}\right)\right) \delta_{c_i, c_j}, \tag{2.46}$$

where $a_{ij},\ b_{ij} \geq 0$ are some weights. The intuition behind this formula is that a community should have more edges inside it than edges which connect it to other communities. Thus, it is necessary to reward existing links inside a community and

penalize links that are missing from a community. The smaller the value of $\mathcal{H}$ is, the more desirable the community structure is. However the minimal value might not be unique.

Given the definition of $\mathcal{H}$, it is possible to describe various existing community detection methods with an appropriate choice of parameters, as well as propose alternative methods. The following models are shown to fit into $\mathcal{H}$: Reichardt and Bornholdt (RB), Arenas, Fernándes, and Gómez (AFG), Ronhovde and Nussinov (RN) as well as the label propagation method. RB approach with a configuration null model also covers the original definition of modularity. The authors also propose a new method called constant Potts model (CPM) by choosing $a_{ij} = w_{ij} - b_{ij}$ and $b_{ij} = \gamma$ where w_{ij} is the weight of the edge between nodes i and j, and γ is a constant. CPM is similar to RB and RN models but is simpler and more intuitive. CPM and RN have local weights and are consequently resolution-limit-free, while RB, AFG, and modularity are not.

CHAPTER 3
MODULARITY DENSITY: A NEW COMMUNITY QUALITY METRIC

In the last decade, the most popular community detection method, proposed by Newman [7], has been to maximize the quality metric known as modularity [5,8–10] over all the possible partitions of a network. This metric measures the difference between the fraction of all edges that are within the actual community and such a fraction of edges that would be inside the community in a randomized graph with the same number of nodes and the same degree sequence. It is widely used to measure the strength of the community structure detected by the community detection algorithms. However, modularity maximization has two opposite yet concurrent problems. In some cases, it tends to split large communities into smaller communities. In other cases, it tends to form large communities by merging communities that are smaller than a certain threshold which depends on the total number of edges in the network and on the degree of inter-connectivity between the communities. The latter problem is known in the literature as the resolution limit problem [11] which we have discussed in detail in Subsection 2.2.3. Moreover, Good et al. [63] showed that the range of modularity values computed over all possible partitions of a graph has a structure in which the maximum modularity partition is typically concealed among an exponentially large (in terms of the graph size) number of structurally dissimilar, high-modularity partitions.

To address this resolution limit problem, multi-resolution versions of modularity [58, 64] were proposed to allow researchers to specify a tunable target resolution limit parameter and identify communities on that scale. Typically, it is not clear how to choose the correct value for this parameter. Furthermore, Lancichinetti and

Portions of this chapter previously appeared as: M. Chen, T. Nguyen, and B. K. Szymanski, "A new metric for quality of network community structure," *ASE Human J.*, vol. 2, no. 4, pp. 226-240, Sep. 2013.

Portions of this chapter previously appeared as: M. Chen, T. Nguyen, and B. K. Szymanski, "On measuring the quality of a network community structure," in *Proc. ASE/IEEE Int. Conf. Social Computing*, Washington, DC, 2013, pp. 122-127.

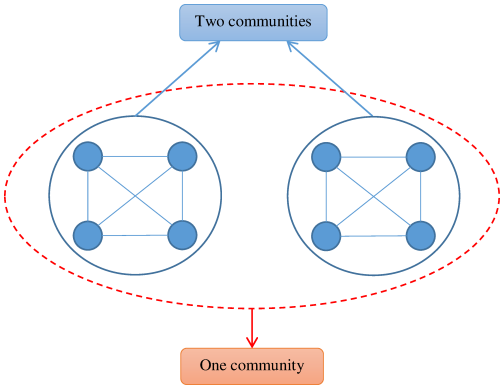
Fortunato [59] stated that even those multi-resolution versions of modularity as well as its original version are not only inclined to merge the smallest well-formed communities but also to split the largest well-formed communities.

To solve these two problems simultaneously, we propose a new community quality metric, that we termed *Modularity Density*, as an alternative to modularity. The modularity density metric eliminates those two problems of modularity without the trouble of specifying any particular parameter. First, we show modularity decreased by *Split Penalty*, defined as the fraction of edges that connect nodes of different communities, avoids the problem of favoring small communities. Next, we demonstrate that including community density into modularity eliminates the problem of favoring large communities. We refer to the resulting metric as modularity density.

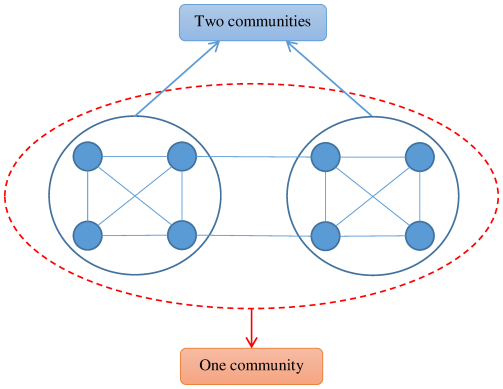
We formally prove that modularity density could resolve the resolution limit problem. We also discuss our experiments with this metric, modularity, and other popular community quality metrics, including the number of *Intra-edges*, *Contraction*, the number of *Inter-edges*, *Expansion*, and *Conductance* [33], on two real-world dynamic networks. The results show that modularity density is different from the original modularity, but consistent with all those quality measures, which implies that modularity density is effective in measuring the community quality of networks.

3.1 Modularity Density

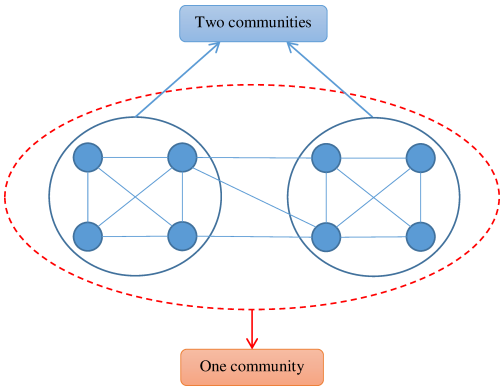
In this section, we first illustrate the motivation for modifying modularity with several simple network examples. Next, we propose a new community quality metric, called *Modularity Density*, as an alternative to modularity by combining modularity with *Split Penalty* and community density to avoid the two coexisting problems of modularity. Finally, we define modularity density for different kinds of networks, including unweighted and undirected networks, weighted networks, and directed networks, based on the corresponding formulas of modularity.



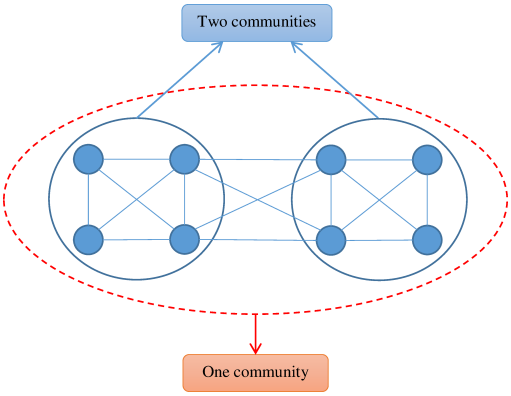
(a) Two very well separated communities.



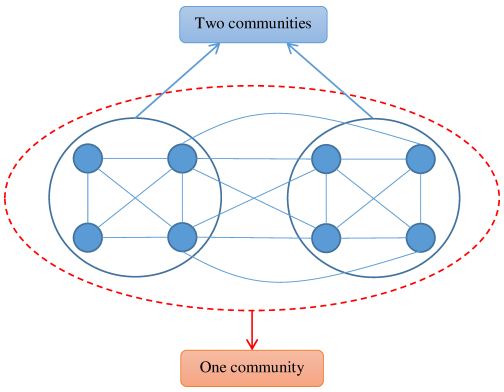
(b) Two well separated communities.



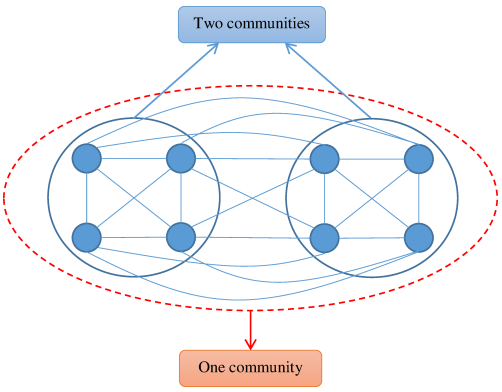
(c) Two weakly connected communities.



(d) Ambiguity between one and two communities.



(e) One well connected community.



(f) One very well connected community.

Figure 3.1: Six simple network examples that have two different community structures, one with a single big community containing all eight nodes and the other with the two small communities each containing four different nodes.

3.1.1 Motivation for Introducing Split Penalty

In this subsection, we demonstrate the motivation for introducing *Split Penalty* into modularity by using seven intuitively clear and simple network examples, six

Table 3.1: Metric values of the example: Two very well separated communities.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	0.5	0	0.5	0.5
One community	0	0	0	0.245

Table 3.2: Metric values of the example: Two well separated communities.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	0.357	0.143	0.214	0.339
One community	0	0	0	0.25

Table 3.3: Metric values of the example: Two weakly connected communities.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	0.3	0.2	0.1	0.263
One community	0	0	0	0.249

of which are presented in Figure 3.1. The seventh example is a complete graph with eight nodes and one big community containing all eight nodes while the alternative partition consists of the two small communities each containing four different nodes. We could easily judge that for the first, second, and the third examples, the community structure with two small communities is better than the community structure in which they are merged together. For the fourth example, the two different community structures are nearly of the same quality. However, for the fifth, sixth, and the seventh examples, the community structure with one big community is of better quality than the alternative.

Tables 3.1-3.7 show the metric values of the seven network examples described above. Tables 3.1-3.3, and Table 3.7 demonstrate that modularity succeeds in measuring the quality of the two different community structure in those four examples. However, from Tables 3.4-3.6, we could observe that modularity actually fails to measure the community quality of those three examples because it implies that the community structure with two small communities is better. In contrast, for the fifth and the sixth examples, the community structure with one big community is of better quality. Yet, in this case modularity gives preference to the community

Table 3.4: Metric values of the example: Ambiguity between one and two communities.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	0.25	0.25	0	0.188
One community	0	0	0	0.245

Table 3.5: Metric values of the example: One well connected community.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	0.167	0.333	-0.167	0.0417
One community	0	0	0	0.23

structure with two separated small communities, demonstrating that modularity has the problem of favoring small communities.

To address the drawback of favoring small communities, we propose that the quality of the community structure should take into account the edges between different communities. We introduce *Modularity with Split Penalty* (Q_s) by subtracting from modularity the *Split Penalty* (SP) which is the fraction of edges that connect nodes of different communities. More formally,

$$Q_s = Q - SP. \tag{3.1}$$

The intuition here is clear. Modularity measures the positive effect of grouping nodes together in terms of taking into account existing edges between nodes while split penalty measures the negative effect of ignoring edges joining members of different communities. Enlarging community eliminates some split penalty but if there are only a few edges across current partition, modularity of the merged community could be lower, negating the benefit of merging. Splitting a community into two or more communities introduces some split penalty but if there are only a few edges between those separated communities, an increase of modularity can make such splitting beneficial. Tables 3.1-3.7 demonstrate that Q_s can correctly measure the quality of the community structure of all seven network examples.

Table 3.6: Metric values of the example: One very well connected community.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	0.0455	0.455	-0.409	-0.239
One community	0	0	0	0.168

Table 3.7: Metric values of the example: One Complete Graph.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two communities	-0.0714	0.571	-0.643	-0.643
One community	0	0	0	0

3.1.2 Modularity with Split Penalty

In this subsection, we extend the formula of Q_s to different kinds of networks, such as unweighted and undirected networks, weighted networks, and directed networks, based on the corresponding formulas of modularity presented in Subsection 2.2.1.

From Subsection 3.1.1, we know that *Split Penalty* (SP) is the fraction of edges that connect nodes of different communities. Thus, for undirected networks, no matter unweighted or weighted, *Split Penalty* is defined as

$$SP = \sum_{c \in C} \left[\sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{2|E|} \right]. \tag{3.2}$$

where $|E_{c,c'}|$ is the number of edges from community c to community c' for unweighted networks or the sum of the weights of the edges from community c to community c' for weighted networks. For directed networks, *Split Penalty* is given by

$$SP = \sum_{c \in C} \left[\sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{|E|} \right]. \tag{3.3}$$

It can be seen that for each community, the split penalty only takes into account the outgoing edges from this community to the rest of the network but not the incoming edges from the rest of the network to this community. It is reasonable to use only outgoing edges, because in a sense those are friendships of community members.

Incoming edges may not be apparent. Moreover, considering both outgoing and incoming edges would only double the value of split penalty because the incoming edges of a community are the outgoing edges of other communities.

Therefore, for undirected networks, both unweighted and weighted, from Equations (2.9), (3.1), and (3.2), Q_s is defined as

$$\begin{aligned} Q_s &= Q - SP \\ &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 \right] - \sum_{c \in C} \left[\sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{2|E|} \right] \\ &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 - \sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{2|E|} \right]. \end{aligned} \tag{3.4}$$

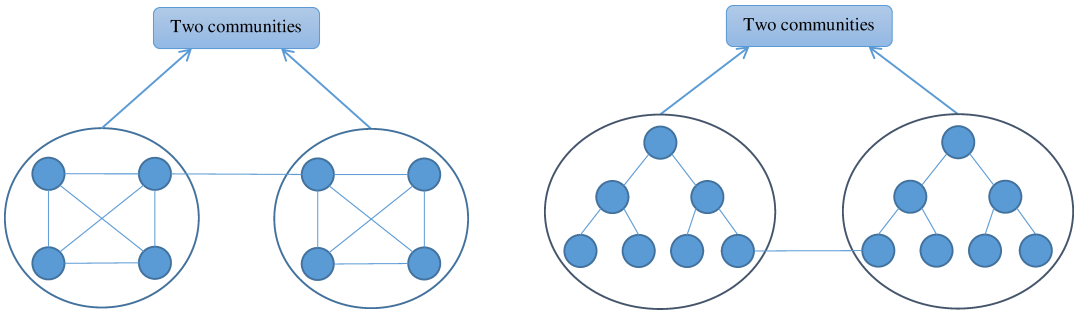
For directed networks, using Equations (2.12), (3.1), and (3.3), Q_s can be expressed as

$$\begin{aligned} Q_s &= Q - SP \\ &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \frac{(|E_c^{in}| + |E_{out,c}|)(|E_c^{in}| + |E_{c,out}|)}{|E|^2} \right] - \sum_{c \in C} \left[\sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{|E|} \right] \\ &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \frac{(|E_c^{in}| + |E_{out,c}|)(|E_c^{in}| + |E_{c,out}|)}{|E|^2} - \sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{|E|} \right]. \end{aligned} \tag{3.5}$$

3.1.3 Motivation for Introducing Community Density

Modularity and also Q_s have two shortcomings. First, they are independent of the number of nodes in the communities as long as the number of edges is preserved. Second, modularity has the resolution limit problem that Q_s makes even worse.

The first shortcoming is illustrated in Figure 3.2 with two simple networks. The left subfigure contains two clique communities and the right subfigure includes two tree communities. In each subfigure, there is one single edge that connects the two communities and there are six edges within all four communities but the number of nodes in clique communities is different from the number of nodes in



(a) Two clique communities. (b) Two tree communities.

Figure 3.2: Two simple network examples with the left one containing two clique communities and the right one containing two tree communities. Also, there are six edges within all four communities, but the number of nodes is different in clique and tree communities.

Table 3.8: Metric values of the example: two clique communities vs two tree communities.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Two clique communities	0.4231	0.07692	0.3462	0.4183
Two tree communities	0.4231	0.07692	0.3462	0.2214

tree communities. As shown in Table 3.8, the values of modularity and Q_s of those two different community structure are the same. However, it is quite obvious that the two clique communities have better community structure quality than the two tree communities in terms of node connections. Moreover, this example shows that the number of nodes of the network and within the communities influences neither modularity nor Q_s.

Second shortcoming, the resolution limit problem, is illustrated in Figure 3.3. It displays a ring network comprised of thirty identical cliques, each of which has five nodes and they are connected by single edges. In this case, the modularity of the community structure with each clique forming a different community, totally thirty communities, should be larger than that of the community structure in which two consecutive cliques form a different community, totally fifteen communities. However, Table 3.9 shows that the relation is reversed since the community structure with fifteen communities has larger modularity than that of the community structure with thirty communities. Further, as pointed out in [11], when $m(m-1)+2 < n$, where n is the number of cliques and m is the number of nodes in each clique,

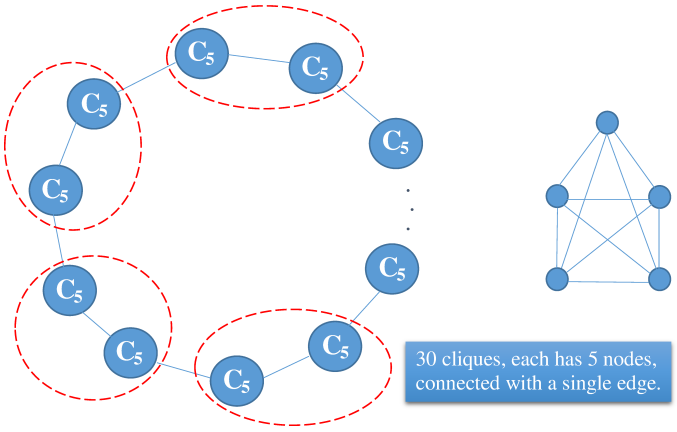


Figure 3.3: A ring network example made out of thirty identical cliques, each having five nodes and connected by single edges.

Table 3.9: Metric values of the example: a ring of thirty cliques, each having five nodes and connected by single edges.

	Modularity (Q)	*Split Penalty* (SP)	Q_s	Q_{ds}
Thirty communities	0.8758	0.09091	0.7848	0.8721
Fifteen communities	0.8879	0.04545	0.8424	0.4305

modularity is higher for the large community with two consecutive cliques instead of the small community with a single clique. Moreover, Table 3.9 demonstrates that the difference of Q_s for these two community structure is larger than the corresponding difference of modularity. More specifically, $\Delta Q_s = (0.8424 - 0.7848) = 0.0576 > \Delta Q = (0.8879 - 0.8758) = 0.0121$, which means that Q_s makes the resolution limit problem even worse.

To address the above two shortcomings, it is quite intuitive to introduce community density into modularity, incorporating both the number of edges and the number of nodes in the communities and also split penalty. The corresponding new metric is called *Modularity Density* (Q_{ds}). Table 3.8 implies that the Q_{ds} of the two tree communities is almost half of the Q_{ds} of the two clique communities. Moreover, Table 3.9 shows that the Q_{ds} of the community structure in which two consecutive cliques form a different community is almost half of the Q_{ds} of the alternative in which each clique forms a different community. Hence, in this case, Q_{ds} avoids the resolution limit problem. Furthermore, Tables 3.1-3.7 and Figure 3.1 demonstrate that Q_{ds} correctly measures the quality of the community structure of all seven

network examples. Even for the network example of Figure 3.1(d) in which there is ambiguity which community structure is of higher quality, the Q_{ds} of the one big community is only slightly larger than the Q_{ds} of the two small communities as shown in Table 3.4.

3.1.4 Modularity Density

In this subsection, we will give the formulas for Q_{ds} for different kinds of networks, including unweighted and undirected networks, weighted networks, and directed networks, based on the corresponding formulas of Q_s presented in Subsection 3.1.2.

For undirected networks, regardless whether unweighted or weighted, we define Q_{ds} using Equation (3.4) as follows

$$\begin{aligned} Q_{ds} &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} d_c - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} d_c \right)^2 - \sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{2|E|} d_{c,c'} \right], \\ d_c &= \frac{2|E_c^{in}|}{|c|(|c|-1)}, \\ d_{c,c'} &= \frac{|E_{c,c'}|}{|c||c'|}. \end{aligned} \tag{3.6}$$

In the above, d_c is the internal density of community c, $d_{c,c'}$ is the pair-wise density between community c and community c'. Note that $|E_c^{in}|$ in d_c and $|E_{c,c'}|$ in $d_{c,c'}$ are unweighted for both unweighted and weighted networks, so that those two community densities are always less than or equal to 1.0.

For directed networks, using Equation (3.5), Q_{ds} is given by

$$\begin{aligned} Q_{ds} &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} d_c - \frac{(|E_c^{in}| + |E_{out,c}|)(|E_c^{in}| + |E_{c,out}|)}{|E|^2} d_c^2 - \sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{|E|} d_{c,c'} \right], \\ d_c &= \frac{|E_c^{in}|}{|c|(|c|-1)}, \\ d_{c,c'} &= \frac{|E_{c,c'}|}{|c||c'|}. \end{aligned} \tag{3.7}$$

3.2 Evaluation and Analysis

In this section, we first prove that modularity density (Q_{ds}) resolves the resolution limit problem. Then, we introduce two real-world dynamic datasets and various other popular community quality measurements. Finally, we show the experimental results that validate Q_{ds} ability to avoid the two problems of modularity (Q) simultaneously.

3.2.1 Proof of Solving Resolution Limit Problem

In this subsection, we test modularity density (Q_{ds}) on the examples from Fortunato and Barthélemy [11]. First, we prove that Q_{ds} does not divide a clique into two or more parts. Then, we verify that Q_{ds} will not merge two or more adjacent cliques connected with a single edge. Finally, we prove that Q_{ds} can discover communities with different sizes.

***Modularity Density* (Q_{ds}) does not divide a clique into two or more parts.** Given a clique with m ($m \geq 3$) nodes, we prove that maximizing Q_{ds} does not divide this clique into two parts. Consider an arbitrary partition P that divides the clique into communities c_1 and c_2 with the number of nodes m_1 and m_2, respectively. Then, the number of edges between c_1 and c_2 is $m_1 m_2$. Let $Q_{ds}(single)$ be the Q_{ds} of the whole clique and $Q_{ds}(pairs)$ be the Q_{ds} of partition P. By definitions,

$$
\begin{aligned}
&Q_{ds}(single) = 0, \\
&Q_{ds}(pairs) = \frac{(m_1 - m_2)^2 - m}{m(m-1)} - \frac{m_1^2 + m_2^2}{m^2},
\end{aligned}
$$

then,

$$
Q_{ds}(pairs) - Q_{ds}(single) = \frac{-2m_1 m_2 - 2m_1 m_2 m}{m^2(m-1)} < 0.
$$

Hence, Q_{ds} will not divide a clique into two parts. A simple generalization of this proof demonstrates that Q_{ds} will not divide a clique into three or more parts.

***Modularity Density* (Q_{ds}) does not merge two or more consecutive cliques in the clique structure ring network.** Given a network, see Figure 3.4(a), comprised of a ring of n (where $n \geq 2$ is an even integer) cliques connected through single edges. Each clique is a complete graph with m ($m \geq 3$)

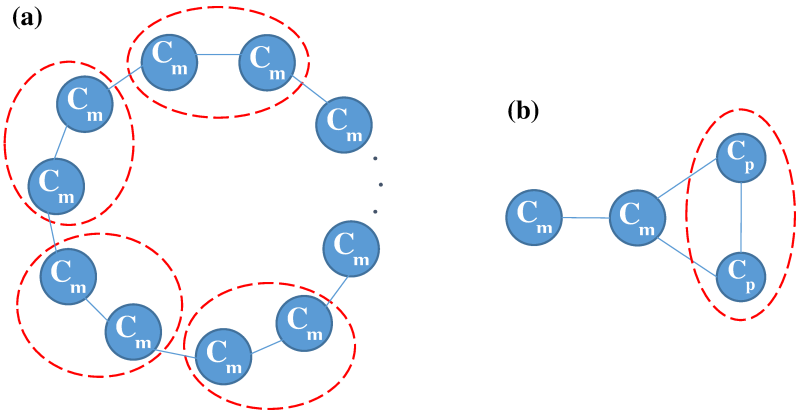


Figure 3.4: Two clique structure network examples. (a) A clique structure ring network. There are totally n (where n is an even positive integer) cliques. Each clique contains m ($m \geq 3$) nodes, and two consecutive cliques are connected by a single edge. (b) A network with two pairs of identical cliques. One pair of cliques have m ($m \geq 4$) nodes, and the other pair of cliques have p ($3 \leq p < m$) nodes.

nodes and $m(m-1)/2$ edges. Then, the cycle network has a total of nm nodes and $nm(m-1)/2+n$ edges. It is clear that the ring network has a well-formed community structure where each community corresponds to a single clique. However, this community structure cannot be obtained by maximizing modularity [11] since the community structure with $n/2$ communities of two adjacent cliques each has higher modularity. We prove that maximizing Q_{ds} finds the right community structure. We let $Q_{ds}(single)$ be the Q_{ds} of the community structure in which each clique is a different community, totally n communities, and $Q_{ds}(pairs)$ be the Q_{ds} of the community structure with two consecutive cliques forming a different community, totally $n/2$ communities. By definitions,

$$Q_{ds}(single) = \frac{m(m-1)}{m(m-1)+2} - \frac{1}{n} - \frac{2}{m^3(m-1)+2m^2},$$

$$Q_{ds}(pairs) = \frac{[m(m-1)+1]^2}{[m(m-1)+2]\,[m(2m-1)]} - \frac{2\,[m(m-1)+1]^2}{n\,[m(2m-1)]^2} - \frac{1}{4m^3(m-1)+8m^2}.$$

We need to prove the inequality

$$Q_{ds}(pairs) < Q_{ds}(single). \tag{3.8}$$

The first term of $Q_{ds}(pairs)$ can be rewritten as

$$\frac{[m(m-1)+1]^2}{[m(m-1)+2][m(2m-1)]} = \frac{m^4-2m^3+3m^2-2m+1}{m(m^2-m+2)(2m-1)}.$$

Then, the first and third terms of $Q_{ds}(single)$ with the latter combined with the last term of $Q_{ds}(pairs)$ yield

$$-\frac{m^2-m}{m^2-m+2} + \frac{7}{4m^2(m^2-m+2)} = -\frac{m^4-m^3-1.75}{m^2(m^2-m+2)}.$$

Combining all these terms, we get

$$\frac{-m^5+m^4+2m^3-2m^2+4.5m-1.75}{m^2(m^2-m+2)(2m-1)}.$$

We move the remaining two terms to the right hand side of Inequality (3.8) that we are proving getting

$$\frac{1}{n}\frac{-2m^4+5m^2-4m+2}{m^2(2m-1)^2}.$$

Multiplying both sides by $-m^2(2m-1)$ (and changing direction of inequality) we get

$$\frac{m^5-m^4-2m^3+2m^2-4.5m+1.75}{m^2-m+2} > \frac{1}{4n}\frac{m^4-2.5m^2+2m-1}{m-0.5}.$$

By doing divisions on both sides, we get

$$m^3-4m-2+\frac{1.5m+5.75}{m^2-m+2} > \frac{1}{4n}\left[m^3+0.5m^2-2.25m+0.875-\frac{9}{16m-8}\right].$$

Since $\frac{1.5m+5.75}{m^2-m+2} \geq 0$, and $\frac{1}{4n} \leq \frac{1}{8}$ for $n \geq 2$ and also $\frac{9}{16m-8} > 0$, we just need to show that

$$m^3-4m-2 > \frac{m^3+0.5m^2-2.25m+0.875}{8}$$

which simplifies to

$$7m^3 - 0.5m^2 - 29.75m - 16.875 > 0 \text{ for } m \geq 3,$$

which is easy to prove either by induction, starting at $m = 3$, or by inspecting zeros of the derivative $21m^2 - m - 29.75$, which are all less than 2.0, showing that this polynomial is positive for $m \geq 3$.

Since Inequality (3.8) holds, Q_{ds} will not merge two consecutive cliques in the ring network. A straightforward extension of the proof shows that Q_{ds} will not merge three or more consecutive cliques.

***Modularity Density* (Q_{ds}) could discover communities with different sizes.** Consider a network, shown in Figure 3.4(b), with two pairs of identical cliques. The left pair of cliques have m ($m \geq 4$) nodes, and the right pair of cliques have p ($3 \leq p < m$) nodes. This network has $2m+2p$ nodes and $m(m-1)+p(p-1)+4$ edges. It is obvious that each of the four cliques should be a different community. However, the authors in [11] found that maximizing modularity will merge the right two small cliques. Here, we prove that maximizing Q_{ds} will not merge them. We let $Q_{ds}(single)$ denote the Q_{ds} of the community structure in which each clique corresponds to a single clique, and $Q_{ds}(pairs)$ be the Q_{ds} of the community structure with the right two small cliques merged into one community. Clearly, the Q_{ds} of the left two large cliques will stay the same in those two different community structure so we denote it as $Q_{ds}(0)$. By definitions,

$$\begin{aligned} Q_{ds}(single) = Q_{ds}(0) &+ \frac{p(p-1)}{m(m-1)+p(p-1)+4} \\ &- \frac{[p(p-1)+2]^2}{2\,[m(m-1)+p(p-1)+4]^2} \\ &- \frac{1}{mp\,[m(m-1)+p(p-1)+4]} \\ &- \frac{1}{p^2\,[m(m-1)+p(p-1)+4]}, \end{aligned}$$

$$Q_{ds}(pairs) = Q_{ds}(0) - \frac{1}{mp\left[m(m-1)+p(p-1)+4\right]} - \frac{\left[p(p-1)+1\right]^2\left[p(p-1)+2\right]^2}{p^2(2p-1)^2\left[m(m-1)+p(p-1)+4\right]^2} + \frac{\left[p(p-1)+1\right]^2}{p(2p-1)\left[m(m-1)+p(p-1)+4\right]}.$$

The inequality that we need to prove is

$$Q_{ds}(single) - Q_{ds}(pairs) > 0. \tag{3.9}$$

Since

$$\begin{aligned} &Q_{ds}(single) - Q_{ds}(pairs) \\ &= \frac{1}{m(m-1)+p(p-1)+4} * \Bigg\{ p(p-1) - \frac{1}{p^2} \\ &- \frac{[p(p-1)+2]^2}{2[m(m-1)+p(p-1)+4]} - \frac{[p(p-1)+1]^2}{p(2p-1)} \\ &+ \frac{[p(p-1)+1]^2[p(p-1)+2]^2}{p^2(2p-1)^2[m(m-1)+p(p-1)+4]} \Bigg\}, \end{aligned}$$

it is clear that the first factor is always positive so it can be removed from consideration and the interior of the second factor can be rewritten as

$$\begin{aligned} &(p^2-p) + \frac{2[p^2-p+1]^2[p^2-p+2]^2 - [p^2-p+2]^2p^2(2p-1)^2}{2p^2(2p-1)^2[m^2-m+p^2-p+4]} \\ &> \frac{1}{p^2} + \frac{[p^2-p+1]^2}{p(2p-1)}. \end{aligned} \tag{3.10}$$

The second term simplifies to

$$-\frac{[p^2-p+2]^2}{2}\frac{2p^4-5p^2+4p-2}{p^2(2p-1)^2[m^2-m+p^2-p+4]}.$$

Since by induction for $p \geq 3$ the polynomial $2p^4 - 5p^2 + 4p - 2$ is positive, then this

term is greater than

$$
\begin{aligned}
&- \frac{(p^2-p+2)(2p^4-5p^2+4p-2)}{4p^2(2p-1)^2} \\
&= \frac{1}{4}\left[-0.5p^2+0.375-\frac{7.5p^3-15.625p^2+10p-4}{4p^4-4p^3+p^2}\right].
\end{aligned}
$$

It is easy to show that the last fraction of the formula above is less than 0.391 by using induction or by finding zeros of the fraction derivative, which are all less than 2.5, so we just need to prove that $0.875p^2-p-0.004$ is greater than the right hand side of Inequality (3.10).

The second term of the right hand side of Inequality (3.10) can be rewritten as

$$
\begin{aligned}
&\frac{p^4-2p^3+3p^2-2p+1}{2(p^2-0.5p)} \\
=&0.5p^2-0.75p+1.125-\frac{0.875p-1}{2(p^2-0.5p)} \\
<&0.5p^2-0.75p+1.125,
\end{aligned}
$$

because $0.875p > 1$ and $p^2 > p$ for $p \geq 2$.

Since $\frac{1}{p^2} < 0.12$, the inequality that we need to prove reduces to $0.375p^2 - 0.25p > 1.249$, but for $p \geq 3$, $0.375p^2 - 0.25p \geq 2.625$, proving Inequality (3.10) and so Inequality (3.9). Thus, we conclude that maximizing Q_{ds} will not merge the right two small cliques, demonstrating that Q_{ds} can discover communities of different size.

In summary, all the above proofs show that in many cases in which Q suffers from limitations our Q_{ds} does not.

3.2.2 Real-world Dynamic Datasets

In this subsection, we introduce two real-world dynamic datasets on which we conduct experiments in order to validate Q_{ds} avoids the two problems of modularity.

Senate Dataset [65, 66]. The Senate dataset is a time-evolving weighted network comprised of United States senators where the weight of an edge represents the similarity of their roll call voting behavior. This dataset was obtained from website *voteview.com* and the similarities between a pair of senators were calculated

following Waugh et al. [66] as the number of bills for which the senators of the pair voted the same way, normalized by the number of bills for which they both voted. The dataset totally consists of 111 snapshots corresponding to Senate's activities over 220 years and includes 1916 unique senators.

Reality Mining Bluetooth Scan Data [67]. This dataset was created from the records of Bluetooth Scans generated among the 94 subjects in Reality Mining study conducted from 2004-2005 at the MIT Media Laboratory. In the network, nodes represent the subjects and the directed edges correspond to the Bluetooth Scan records while the weight of each edge represents the number of direct Bluetooth scans between the two subjects. In the experiments, we only used the records from August 02, 2004 (Monday) to May 29, 2005 (Sunday) and we divided them into weekly snapshots, so each snapshot represents scans collected during the corresponding week. There are total of 43 snapshots.

3.2.3 Other Community Quality Metrics

In the discussion of the experimental results we use various community quality metrics, including the number of *Intra-edges*, *Contraction*, the number of *Inter-edges*, *Expansion*, and *Conductance* [33], which characterize how community-like is the connectivity structure of a given set of nodes. All of them rely on the intuition that communities are sets of nodes with many edges inside them and few edges outside of them. Now, given a network $G = (V, E)$ and given a community or a set of nodes c, let $|c|$ be the number of nodes in the community c and let $|E_c^{in}|$ denote the total number of edges in c for unweighted networks or the total weight of such edges for weighted networks. We denote the total number of edges from the nodes in community c to the nodes outside c for unweighted networks or the total weight of such edges for weighted networks as $|E_c^{out}|$. Then, the definitions of the five quality metrics are as follows:

The number of *Intra-edges*: $|E_c^{in}|$; it is the total number of edges in c or the total weight of such edges. A large value of this metric is better than a small value in terms of the community quality.

***Contraction*:** $2|E_c^{in}|/|c|$ for undirected networks or $|E_c^{in}|/|c|$ for directed networks;

it measures the average number of edges per node inside the community c or the average weight per node of such edges. A large value of *Contraction* is better than a small value in terms of the community quality.

The number of *Inter-edges*: $|E_c^{out}|$; it is the total number of edges from the nodes in community c to the nodes outside c or the total weight of such edges. A small value of this metric is better than a large value in terms of the community quality.

***Expansion*:** $|E_c^{out}|/|c|$; it measures the average number of edges (per node) that point outside community c or the average weight per node of such edges. A small value of *Expansion* is better than a large value in terms of the community quality.

***Conductance*:** $\frac{|E_c^{out}|}{2|E_c^{in}|+|E_c^{out}|}$ for undirected networks or $\frac{|E_c^{out}|}{|E_c^{in}|+|E_c^{out}|}$ for directed networks; it measures the fraction of the total number of edges that point outside the community for unweighted networks or the fraction of the total weight of such edges for weighted networks. A small value of *Conductance* is better than a large value in terms of the community quality.

3.2.4 Experimental Results

In this subsection, we report the results of performing community detection on the two real-world dynamic datasets introduced in Subsection 3.2.2 by using the dynamic community detection algorithms, LabelRankT [68] and Estrangement [65]. We chose these two algorithms because the second algorithm relies on the modularity optimization while the first one does not. In the experiments, we adopted the best parameter of Estrangement but varying the conditional update parameter $q \in [0, 1]$ of LabelRankT from 0.05 to 0.95. As seen in the results, in most cases, the best q is around 0.7 in agreement with the best value reported in [68]. For the community structure found by the two algorithms, we calculated the values of modularity (Q), Q_s, modularity density (Q_{ds}), and the five community quality metrics described in Subsection 3.2.3.

Table 3.10 and Table 3.11 present the average metric differences between LabelRankT with different values of conditional update parameter q and Estrangement on Senate dataset and Reality Mining Bluetooth Scan data, respectively. That is,

Table 3.10: The average metric differences between LabelRankT with different values of conditional update parameter q and Estrangement on Senate dataset.

LabelRankT q	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95
Q	-0.0534	-0.0462	**-0.0408**	-0.0538	-0.0714	-0.0848	-0.083	-0.0897	-0.0897	-0.0848	-0.08
Q_s	-0.166	-0.0802	0.0468	0.0808	0.0969	0.112	**0.116**	0.115	0.115	0.111	0.106
Q_{ds}	-0.1638	-0.0787	0.04847	0.08297	0.0995	0.1145	0.1182	**0.1183**	**0.1183**	0.1135	0.1083
# Intra-edges	-159.102	-32.444	234.296	387.38	510.645	616.855	615.123	**624.764**	**624.764**	602.627	580.733
Contraction	-6.806	-3.023	2.481	4.553	5.937	7.033	7.065	**7.227**	**7.227**	6.927	6.622
# Inter-edges	-75.962	-54.098	-123.898	-187.99	-245.198	-299.356	-300.108	**-303.043**	**-303.043**	-292.782	-282.442
Expansion	6.448	2.91	-2.428	-4.416	-5.737	-6.847	-6.878	**-7.009**	**-7.009**	-6.724	-6.431
Conductance	0.213	0.0851	-0.0886	-0.148	-0.186	-0.214	-0.216	**-0.224**	**-0.224**	-0.213	-0.201

Table 3.11: The average metric differences between LabelRankT with different values of conditional update parameter q and Estrangement on reality mining bluetooth scan data.

LabelRankT q	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95
Q	-0.161	-0.121	-0.0783	-0.0744	-0.0724	**-0.0699**	-0.0702	-0.0724	-0.0742	-0.0755	-0.0774
Q_s	-0.379	-0.244	-0.107	-0.0802	-0.0538	-0.0497	**-0.0382**	-0.0405	-0.0521	-0.0634	-0.0713
Q_{ds}	-0.191	-0.0984	-0.0222	-0.017	-0.0116	-0.0116	**-0.00318**	-0.00826	-0.011	-0.0115	-0.0134
# Intra-edges	-1450.893	-956.006	-479.377	-331.371	-230.263	-183.536	-102.94	**-78.93**	-155.183	-242.287	-333.419
Contraction	-86.909	-69.914	-52.543	-46.371	-43.176	-40.567	**-35.948**	-36.425	-38.006	-41.277	-45.425
# Inter-edges	-39.949	-76.524	-159.74	-167.333	-190.947	-190.865	**-196.098**	-193.123	-188.708	-179.653	-178.96
Expansion	52.529	25.829	6.289	5.76	5.664	7.07	**4.881**	6.799	6.916	6.117	5.669
Conductance	0.23	0.176	0.114	0.1	0.0934	0.0933	**0.0843**	0.0955	0.102	0.107	0.104

we first computed the values of the eight metrics above for the community detection results, detected by Estrangement, of each snapshot. Then, we calculated the eight metrics values for the community detection results, discovered by LabelRankT for all q, of each snapshot. Next, we got the metric differences of all eight metrics by subtracting the metric values of Estrangement from those of LabelRankT for all q's over each snapshot. Then, averaging those differences of each metric over all the snapshots, we obtained the corresponding average metric differences.

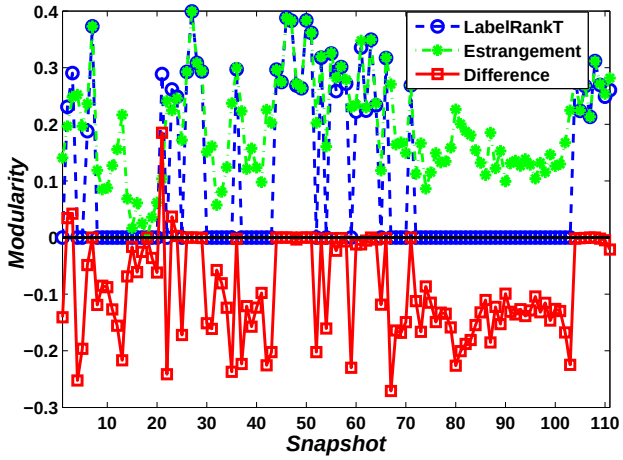
Table 3.10 demonstrates that Q gets its largest value when $q = 0.2$; Q_s reaches the largest value when $q = 0.6$; Q_{ds}, *Intra-edges*, and *Contraction* get their largest values at $q = 0.7$ and $q = 0.8$; also, *Inter-edges*, *Expansion*, and *Conductance* reach their smallest values at $q = 0.7$ and $q = 0.8$. Thus, Q_{ds} is consistent with the five metrics introduced in Subsection 3.2.3 on determining the best q for LabelRankT on Senate dataset while Q and Q_s are not consistent with them. Further, we could observe that Q is always negative which indicates that LabelRankT performs below Estrangement over all q's because the goal of Estrangement is to maximize modularity (Q). However, the other seven metrics imply that LabelRankT performs

better than Estrangement when $q > 0.1$. Therefore, we could explicitly observe that maximizing Q to detect communities has problems in measuring the community detection quality correctly on Senate dataset.

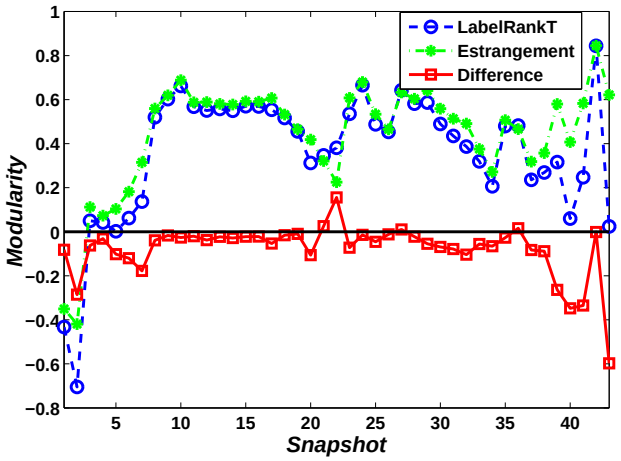
Table 3.11 shows that six quality metrics get their best (largest or smallest) values at $q = 0.6$ while the two exceptions, Q and the number of *Intra-edges*, reach their largest values when $q = 0.5$ and $q = 0.7$, respectively. Thus, the six metrics, except Q and the number of *Intra-edges*, are consistent on determining the best value of q for LabelRankT on Reality Mining Bluetooth Scan data. This indicates that on Reality Mining Bluetooth Scan data, maximizing Q to detect communities has problems.

It is also interesting to observe that for $q = 0.05$ and $q = 0.1$ in Table 3.10, *Inter-edges* metric implies that LabelRankT performs better than Estrangement on Senate dataset, which is not consistent with Q_s, Q_{ds}, *Intra-edges*, *Contraction*, *Expansion*, and *Conductance* metrics. Moreover, we could learn from Table 3.11 that all the metrics, except *Inter-edges* metric, imply that LabelRankT performs slightly below the performance of Estrangement over all q's. Thus, *Inter-edges* metric has some problems. Also, as mentioned in the paragraph above, *Intra-edges* metric is not consistent with the other six metrics on determining the best q for LabelRankT, which also means that *Intra-edges* metric has problems. We conjecture that the reason for the shortcoming of *Intra-edges* and *Inter-edges* metrics is the same as the case of modularity (Q) which does not consider the number of nodes in the communities. This reason also implies the superiority of Q_{ds} over Q and Q_s.

Based on the results presented in the above two tables, we conclude that Q_{ds} eliminates the two problems of modularity. We also conjecture that the difference between the best values of q for LabelRankT determined by Q and Q_s and the difference determined by Q_s and Q_{ds} on Senate dataset is a manifestation of the two problems of modularity maximization, namely favoring small communities and the resolution limit problem. Moreover, the difference between the best values of q for LabelRankT determined by Q and Q_s on Reality Mining Bluetooth Scan data indicates that maximizing Q has the problem of favoring small communities. Thus, Q_s and Q_{ds} can be used for checking whether finding communities by maximizing



(a) Senate dataset ($q = 0.7$).

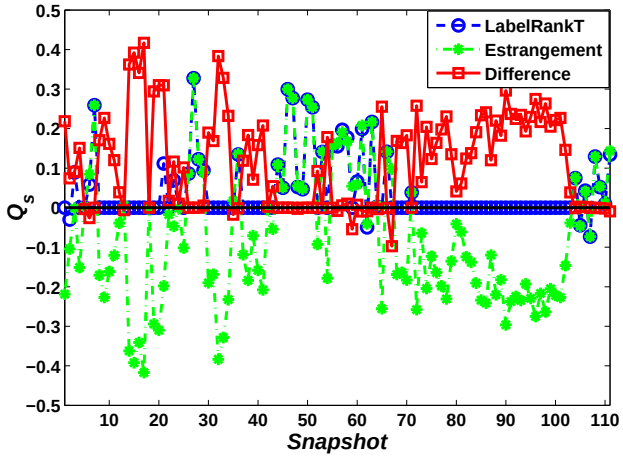


(b) Reality Mining Bluetooth Scan data ($q = 0.6$).

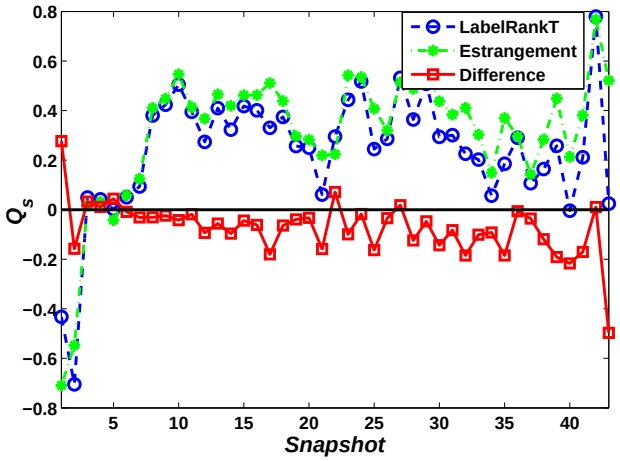
Figure 3.5: The modularity (Q) of the community detection results of LabelRankT and Estrangement (also, the difference between LabelRankT and Estrangement) on (a) each snapshot of Senate dataset at $q = 0.7$ and on (b) each snapshot of Reality Mining Bluetooth Scan data with $q = 0.6$.

Q on a specific dataset will suffer any of the two problems.

To make the differences among Q, Q_s, and Q_{ds} more clear, we plot their values, in Figures 3.5, 3.6, and 3.7, of the community detection results of LabelRankT and Estrangement on each snapshot of Senate dataset at $q = 0.7$ and on each snapshot of Reality Mining Bluetooth Scan data when $q = 0.6$. Figure 3.5(a) shows that in most cases Q is negative, while Q_s and Q_{ds} are positive as seen in Figure 3.6(a) and Figure 3.7(a). It indicates that there is large difference between Q and Q_s or between Q and Q_{ds}. This is consistent with Table 3.10. Further, it can be observed from Figure 3.6(a) and Figure 3.7(a) that Q_s and Q_{ds} are almost the same on each



(a) Senate dataset ($q = 0.7$).



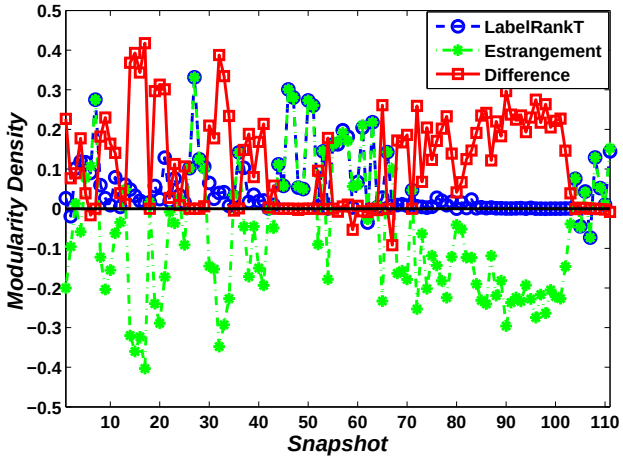
(b) Reality Mining Bluetooth Scan data ($q = 0.6$).

Figure 3.6: Q_s of the community detection results of LabelRankT and Estrangement (also, the difference between LabelRankT and Estrangement) on (a) each snapshot of Senate dataset at $q = 0.7$ and on (b) each snapshot of Reality Mining Bluetooth Scan data with $q = 0.6$.

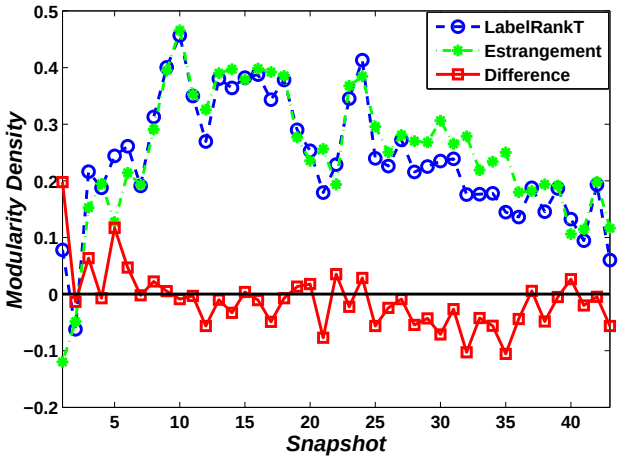
snapshot, which is also consistent with Table 3.10. Figure 3.5(b), Figure 3.6(b), and Figure 3.7(b) demonstrate that Q, Q_s, and Q_{ds} are negative in most of the cases, although their values are different in each snapshot. These observations are consistent with the results shown in Table 3.11.

3.3 Summary

In this chapter, we propose a new community quality metric, called modularity density, which resolves the problems of modularity of favoring small communities in some circumstances and large communities in others. We demonstrate with proof-



(a) Senate dataset ($q = 0.7$).



(b) Reality Mining Bluetooth Scan data ($q = 0.6$).

Figure 3.7: The modularity density (Q_{ds}) of the community detection results of LabelRankT and Estrangement (also, the difference between LabelRankT and Estrangement) on (a) each snapshot of Senate dataset at $q = 0.7$ and on (b) each snapshot of Reality Mining Bluetooth Scan data with $q = 0.6$.

s and experiments on real-world dynamic datasets that modularity density is an effective alternative to modularity.

CHAPTER 4
FUZZY OVERLAPPING COMMUNITY QUALITY METRICS

Newman's modularity [5, 8–10] introduced in Subsection 2.2.1 can only be used to measure the quality of disjoint communities. However, it is more realistic to expect that nodes in real-world networks belong to more than one community, resulting in overlapping communities [12]. Therefore, several overlapping extensions of modularity ([13–19]) were proposed to measure the quality of overlapping community structure. Yet, to date no attempt has been made to systematically compare different overlapping extensions and propose metric selection criteria for different types of networks. In this chapter, we consider several overlapping extensions of modularity and test their quality on real-world and synthetic networks. We also extend localized modularity [20], modularity density [21, 22], and eight local community quality metrics for overlapping communities following the same principles used by the overlapping extensions of modularity.

We conducted experiments on a large number of real-world networks and synthetic networks using overlapping extensions of modularity, overlapping modularity density, and eight local metrics (the number of Intra-edges, Intra-density, Contraction, the number of Boundary-edges, Expansion, Conductance [21, 22, 69], the Fitness function [70], and the Average Modularity Degree [71]). The results show that selecting the product of the belonging coefficients of two nodes as a belonging function for overlapping extensions yields better results on these networks than using other belonging functions. The experimental results also give a guidance to researchers on which metrics to choose when measuring the quality of overlapping community structure.

Portions of this chapter previously appeared as: M. Chen, K. Kuzmin, and B. K. Szymanski, "Extension of modularity density for overlapping community structure," in *Proc. 2014 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining*, Beijing, China, 2014, pp. 856-863.

Portions of this chapter previously appeared as: M. Chen and B. K. Szymanski, "Fuzzy overlapping community quality metrics," *Soc. Netw. Anal. Min.*, vol. 5, no. 1, pp. 1-14, Jul. 2015.

Methodology: Below we introduce in steps the methodology we use to evaluate overlapping extensions of modularity:

(1) We first give a generalized definition for existing overlapping extensions of modularity which covers four such extensions, each using one of the two different versions of belonging coefficient and one of the two different versions of belonging function.

(2) Next, we extend localized modularity, modularity density, and eight local community quality metrics to be applicable to overlapping community structure following the same principle as the overlapping extensions of modularity do.

(3) Then, for the generalized definitions of these metrics, we first determine which version of the belonging coefficient and which version of the belonging function perform best for each metric.

(4) Moreover, we determine which version of the belonging coefficient and which version of the belonging function scores the largest number of quality metrics consistent with each other on determining the best values of parameters of compared community detection algorithms: the threshold r for SLPA [72, 73], the parameter k for CFinder [23], and the threshold tr for SpeakEasy [74] on the real-world and synthetic networks.

(5) Finally, we compare the performance of the overlapping metrics with the best combination of belonging coefficient and belonging function by looking at how many times each metric is among those that are consistent with each other on determining the best values of parameters for the same algorithms as used in step (4).

4.1 Overlapping Definition of Modularity

Newman's modularity is used to measure the quality of disjoint community structure of a network. However, it is more realistic that nodes in networks belong to more than one community, resulting in overlapping communities [12]. For

instance, a researcher may be active in several research areas, and a node in biological networks might have multiple functions. It is also quite common that people in social networks are naturally characterized by multiple community memberships depending on their families, friends, professional colleagues, neighbors, etc. For this reason, discovering overlapping communities became very popular in the last few years. Several overlapping extensions of modularity [13–19] were proposed to measure the quality of overlapping community structure. These extensions are described below.

If communities overlap, each node can belong to multiple communities, although the strength of this connection can generally be different for different communities. Given a set of overlapping communities $C = \{c_1, c_2, ..., c, ..., c_{|C|}\}$ in which a node may belong to more than one of them, a vector of *belonging coefficients* $(a_{i,c_1}, a_{i,c_2}, ..., a_{i,c}, ..., a_{i,c_{|C|}})$ [14,18] can be assigned to each node i in the network. $|C|$ is the number of communities. The belonging coefficient $a_{i,c}$ measures the strength of association between node i and community c. Without loss of generality, the following constraints are assumed to hold:

$$\begin{gathered} 0 \leq a_{i,c} \leq 1 \quad \forall i \in V, \forall c \in C \\ \text{and} \\ \sum_{c \in C} a_{i,c} = 1. \end{gathered} \tag{4.1}$$

Therefore, the belonging strength is measured as a real value in the range of $[0, 1]$ and the sum of all belonging coefficients, which is 1, is the same for all nodes in the network.

Zhang et al. [13] proposed an extended modularity which uses the average of the belonging coefficients of two nodes as belonging function to measure the quality of overlapping community structure:

$$Q_{ov}^{Z} = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 \right], \tag{4.2}$$

where $|E_c^{in}| = \frac{1}{2} \sum_{i,j \in c} \frac{a_{i,c}+a_{j,c}}{2} A_{ij}$, $|E_c^{out}| = \sum_{i \in c, j \in V-c} \frac{a_{i,c}+(1-a_{j,c})}{2} A_{ij}$, and $|E| =$

$\frac{1}{2}\sum_{ij} A_{ij}$. For the case of disjoint communities, Q_{ov}^{Z} reduces exactly to Newman's modularity (Q) given by Equation (2.9).

Nepusz et al. [14] considered the belonging coefficient $a_{i,c}$ as the probability that node i is active in community c. Then, the probability that node i is active in the same communities as node j is the dot product of their membership vectors, denoted as s_{ij}:

$$s_{ij} = \sum_{c \in C} a_{i,c} a_{j,c}. \tag{4.3}$$

The authors also adopted s_{ij} as the similarity measure between nodes i and j. By replacing δ_{c_i,c_j} in Equation (2.10) with the similarity measure s_{ij} defined above, they proposed a fuzzified variant of modularity:

$$\begin{aligned} Q_{ov}^{F} &= \frac{1}{2|E|} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] s_{ij} \\ &= \frac{1}{2|E|} \sum_{c \in C} \sum_{i,j \in c} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] a_{i,c} a_{j,c}. \end{aligned} \tag{4.4}$$

In case communities are disjoint, there exists only one community c for every node i for which $a_{i,c} = 1$. Then, the fuzzified modularity (Q_{ov}^{F}) reduces to exactly the original modularity (Q) described in Equation (2.10).

Shen et al. [15] proposed an extension of modularity for overlapping community structure using Equation (4.4) but defined the belonging coefficients of node i to be the reciprocal of the number of communities to which it belongs:

$$a_{i,c} = \frac{1}{O_i}, \tag{4.5}$$

where O_i is the number of communities containing node i. Then, the extended modularity for overlapping community structure is given by:

$$\begin{aligned} Q_{ov}^{E} &= \frac{1}{2|E|} \sum_{c \in C} \sum_{i,j \in c} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] a_{i,c} a_{j,c} \\ &= \frac{1}{2|E|} \sum_{c \in C} \sum_{i,j \in c} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] \frac{1}{O_i O_j}. \end{aligned} \tag{4.6}$$

For disjoint community structure, Q_{ov}^{E} reduces to the original modularity (Q) de-

scribed in Equation (2.10).

Shen et al. [16] proposed another extension of modularity for overlapping communities also using Equation (4.4). In this case, the coefficient of node i belonging to community c is defined as:

$$a_{i,c} = \frac{1}{a_i} \sum_{k \in c} \frac{M_{ik}^c}{M_{ik}} A_{ik}, \tag{4.7}$$

where M_{ik} denotes the number of maximal cliques in the network containing edge (i, k), M_{ik}^c is the number of maximal cliques in community c that contains edge (i, k), and a_i is a normalization term defined as:

$$a_i = \sum_{c \in C} \sum_{k \in c} \frac{M_{ik}^c}{M_{ik}} A_{ik}. \tag{4.8}$$

The maximal clique is a clique that is not a subset of any other cliques. Then, the extended modularity for overlapping community structure is given by:

$$Q_{ov}^C = \frac{1}{2|E|} \sum_{c \in C} \sum_{i,j \in c} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] a_{i,c} a_{j,c}. \tag{4.9}$$

Note that for disjoint communities, this new extension also reduces to Newman's modularity shown in Equation (2.10).

Chen et al. [17] also proposed another extension of modularity with the same Equation (4.4) but with the belonging coefficient defined as:

$$a_{i,c} = \frac{\sum_{k \in c} A_{ik}}{\sum_{c' \in C_i} \sum_{k \in c'} A_{ik}}, \tag{4.10}$$

where C_i is the set of communities to which node i belongs. It measures how tightly node i connects to community c. Consequently, the extended definition of

modularity for overlapping community structure is given by:

$$\begin{aligned} Q_{ov}^{O} &= \frac{1}{2|E|}\sum_{c\in C}\sum_{i,j\in c}\left[A_{ij}-\frac{k_i k_j}{2|E|}\right]a_{i,c}a_{j,c} \\ &= \frac{1}{2|E|}\sum_{c\in C}\sum_{i,j\in c}\left[A_{ij}-\frac{k_i k_j}{2|E|}\right]\frac{\sum\limits_{k\in c}A_{ik}}{\sum\limits_{c'\in C_i}\sum\limits_{k\in c'}A_{ik}}\frac{\sum\limits_{k\in c}A_{jk}}{\sum\limits_{c'\in C_j}\sum\limits_{k\in c'}A_{jk}}. \end{aligned} \tag{4.11}$$

Still, for disjoint community structure, Q_{ov}^{O} reduces to the original modularity given by Equation (2.10).

Unlike the node-based extensions of modularity presented above, Nicosia et al. [18] proposed an edge-based extension of modularity for overlapping communities. In this case, the belonging coefficients represent how edges are assigned to communities. The coefficient for edge $l=(i,j)$ belonging to community c is $\beta_{l(i,j),c}=F(a_{i,c},a_{j,c})$, where $F(a_{i,c},a_{j,c})$ could be any function (product, average, or maximum) of $a_{i,c}$ and $a_{j,c}$. After trying several different functions, the authors stated that the best F is a two-dimensional logistic function:

$$F(a_{i,c},a_{j,c})=\frac{1}{(1+e^{-f(a_{i,c})})(1+e^{-f(a_{j,c})})}, \tag{4.12}$$

where $f(a_{i,c})$ is a simple linear scaling function $f(x)=2px-p, p\in R$. In papers [12,75], p was selected to be 30. Then, the expected belonging coefficient of any edge $l=(i,k)$ starting from node i in community c is given by $\beta^{e}_{l(i,k),c}=\frac{1}{|V|}\sum_{k\in V}\beta_{l(i,k),c}$ running over all nodes in the network. Accordingly, the expected belonging coefficient of any edge $l=(k,j)$ pointing to node j in community c is defined as $\beta^{e}_{l(k,j),c}=\frac{1}{|V|}\sum_{k\in V}\beta_{l(k,j),c}$. Then, the edge-based extension of modularity is given by:

$$\begin{aligned} Q_{ov}^{L} &= \frac{1}{2|E|}\sum_{c\in C}\sum_{i,j\in c}\left[r_{ijc}A_{ij}-s_{ijc}\frac{k_i k_j}{2|E|}\right] \\ &= \frac{1}{2|E|}\sum_{c\in C}\sum_{i,j\in c}\left[\beta_{l(i,j),c}A_{ij}-\frac{\beta^{e}_{l(i,k),c}k_i\beta^{e}_{l(k,j),c}k_j}{2|E|}\right], \end{aligned} \tag{4.13}$$

where

$$r_{ijc}=\beta_{l(i,j),c}=F(a_{i,c},a_{j,c}) \tag{4.14}$$

and

$$\begin{aligned} s_{ijc} &= \beta^{e}_{l(i,k),c}\beta^{e}_{l(k,j),c} \\ &= \frac{\sum_{k\in V}\beta_{l(i,k),c}\sum_{k\in V}\beta_{l(k,j),c}}{|V|^2} \\ &= \frac{\sum_{k\in V}F(a_{i,c},a_{k,c})\sum_{k\in V}F(a_{k,c},a_{j,c})}{|V|^2}. \end{aligned} \tag{4.15}$$

In Q^L_{ov}, r_{ijc} is used as the weight corresponding to the probability of the observed edge $l = (i, j)$, while s_{ijc} is used as the weight of the probability of an edge from node i to node j in the null model. Note that although for disjoint communities $F(a_{i,c}, a_{j,c})$ is practically 1 when both $a_{i,c}$ and $a_{j,c}$ are equal to 1, Q^L_{ov} does not exactly reduce to the original modularity given by Equation (2.10).

Generally, there are two categories of overlapping community structure: crisp (non-fuzzy) overlapping and fuzzy overlapping [19]. For crisp overlapping community structure, each node belongs to one or more communities but without the corresponding belonging coefficients. That is, the relationship between a node and a community is binary: a node either belongs to a community or it does not. For fuzzy overlapping community structure, each node can be a member of multiple communities, but in general the values of belonging coefficients are different. Fuzzy overlapping can be easily transformed to crisp overlapping with a threshold parameter. Namely, if the belonging coefficient of node i to community c is larger than the value of the threshold, then node i stays in community c. Otherwise, node i is deleted from community c. Crisp overlapping can be converted to fuzzy overlapping by calculating the value of the belonging coefficient using Equations (4.5), (4.7), or (4.10). However, calculating the belonging coefficient using Equation (4.7) is computationally expensive since it needs to find all the maximal cliques of the network first. Hence, in this chapter we only consider Equation (4.5) and Equation (4.10) when converting crisp overlapping to fuzzy overlapping.

Now, we give two general definitions, Q_{ov} and Q'_{ov}, for node-based extensions

of modularity. First, Q_{ov} is given by:

$$Q_{ov} = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 \right], \tag{4.16}$$

where $|E_c^{in}| = \frac{1}{2} \sum_{i,j \in c} f(a_{i,c}, a_{j,c}) A_{ij}$, $|E_c^{out}| = \sum_{i \in c} \sum_{\substack{c' \in C \\ c' \neq c \\ j \in c'}} f(a_{i,c}, a_{j,c'}) A_{ij}$, and $|E| = \frac{1}{2} \sum_{ij} A_{ij}$. The belonging function $f(a_{i,c}, a_{j,c})$ can be the average or product of $a_{i,c}$ and $a_{j,c}$. That is, $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c}+a_{j,c}}{2}$ or $f(a_{i,c}, a_{j,c}) = a_{i,c}a_{j,c}$. Clearly, Q_{ov} with $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c}+a_{j,c}}{2}$ is very similar to Q_{ov}^Z in Equation (4.2). Second, Q'_{ov} is given by:

$$Q'_{ov} = \frac{1}{2|E|} \sum_{c \in C} \sum_{i,j \in c} \left[A_{ij} - \frac{k_i k_j}{2|E|} \right] f(a_{i,c}, a_{j,c}). \tag{4.17}$$

where $f(a_{i,c}, a_{j,c})$ is the same as that in Equation (4.16). It is worth noting that Q'_{ov} with the belonging function $f(a_{i,c}, a_{j,c}) = a_{i,c}a_{j,c}$ is actually the same as Q_{ov}^F in Equation (4.4), Q_{ov}^E in Equation (4.6), Q_{ov}^C in Equation (4.9), and Q_{ov}^O in Equation (4.11). The only difference between these formulas is how the value of $a_{i,c}$ is calculated.

It is easy to prove that Q_{ov} is equivalent to Q'_{ov} when $f(a_{i,c}, a_{j,c}) = a_{i,c}a_{j,c}$. From the definition of Q_{ov}, we know that $|E_c^{in}| = \frac{1}{2} \sum_{i,j \in c} a_{i,c}a_{j,c}A_{ij}$ which is in fact the same as the first term of Q'_{ov}. Moreover, it is easy to show that $(2|E_c^{in}| + |E_c^{out}|)^2 = \sum_{i,j \in c} k_i k_j a_{i,c}a_{j,c}$ (see APPENDIX A.1). Hence, the second term of Q_{ov} is the same as the second term of Q'_{ov}. Similarly, it can be shown that Q_{ov} is not equal to Q'_{ov} when $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c}+a_{j,c}}{2}$.

4.2 Localized Modularity Based on Community's Neighborhood

Newman's modularity is a global measure which assumes that all pairs of nodes have equal probability to connect with each other, which reflects the connectivity among all communities. However, Muff et al. [20] argued that in many complex networks most communities are connected to only a small fraction of remaining communities, called local cluster connectivity property. Thus, they modified the

definition of Newman's modularity by taking into account local cluster connectivity only to overcome global network dependency. The resulting measure is called localized modularity. We denote the localized modularity here as NQ because it is based on the neighborhood of a community. NQ for unweighted and undirected networks is given by:

$$NQ = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E_c^{neighb}|} - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E_c^{neighb}|} \right)^2 \right], \tag{4.18}$$

where $|E_c^{neighb}|$ is the total number of edges in the subnetwork containing the community c and all its neighboring communities, i.e. the neighborhood of community c. It means that the contribution of each community c to NQ is calculated based on the neighborhood of c.

Unlike the traditional modularity (Q), the localized version of modularity (NQ) is not bounded above by 1. The more locally connected communities a network has, the bigger its NQ can grow. In a network where all communities are connected to each other, NQ yields the same value as Q. NQ considers individual communities and their neighbors, and therefore provides a measure of community quality that is not dependent on other parts of the network.

Similar to the general node-based overlapping definition of modularity Q_{ov} in Equation (4.16), we extend NQ for overlapping community structure as NQ_{ov}. The formula for NQ_{ov} is exactly the same with NQ in Equation (4.18), while the difference is that $|E_c^{in}|$, $|E_c^{out}|$, and $|E_c^{neighb}|$ in NQ_{ov} should consider the belonging coefficients of nodes and also the belonging function between pairs of nodes. For disjoint communities, NQ_{ov} reduces exactly to NQ.

4.3 Modularity Density for Overlapping Communities

According to Q_{ov} in Equation (4.16), we extend Q_{ds} for overlapping community structure as:

$$
\begin{aligned}
Q_{ds}^{ov} &= \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} d_c - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} d_c \right)^2 - \sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{2|E|} d_{c,c'} \right], \\
d_c &= \frac{2|E_c^{in}|}{\sum_{i,j \in c, i \neq j} f(a_{i,c}, a_{j,c})}, \\
d_{c,c'} &= \frac{|E_{c,c'}|}{\sum_{i \in c, j \in c'} f(a_{i,c}, a_{j,c'})},
\end{aligned}
\tag{4.19}
$$

where in the formula $|E_c^{in}| = \frac{1}{2} \sum_{i,j \in c} f(a_{i,c}, a_{j,c}) A_{ij}$, $|E_c^{out}| = \sum\limits_{i \in c} \sum\limits_{\substack{c' \in C \\ c' \neq c \\ j \in c'}} f(a_{i,c}, a_{j,c'}) A_{ij}$, $|E_{c,c'}| = \sum_{i \in c, j \in c'} f(a_{i,c}, a_{j,c'}) A_{ij}$, and $|E| = \frac{1}{2} \sum\limits_{ij} A_{ij}$. The belonging function $f(a_{i,c}, a_{j,c})$ can be the product or average of $a_{i,c}$ and $a_{j,c}$. For disjoint communities, Q_{ds}^{ov} reduces exactly to Q_{ds} given by Equation (3.6). Notice that we do not extend modularity density based on Q_{ov}^{L} since it is too complicated and far from intuitive.

4.4 Evaluation and Analysis

From Subsection 4.1, we know that all node-based overlapping extensions of modularity can be expressed with Q_{ov} in Equation (4.16) using the belonging function $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c} + a_{j,c}}{2}$ or $f(a_{i,c}, a_{j,c}) = a_{i,c} a_{j,c}$. For the edge-based overlapping extension of modularity (Q_{ov}^{L}), the belonging function is given by Equation (4.12). Also, the overlapping extension of the localized modularity NQ_{ov} has the belonging function $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c} + a_{j,c}}{2}$ or $f(a_{i,c}, a_{j,c}) = a_{i,c} a_{j,c}$. For the overlapping extension of modularity density (Q_{ds}^{ov}), the belonging function $f(a_{i,c}, a_{j,c})$ can also be the average or the product of $a_{i,c}$ and $a_{j,c}$. Thus, there are two versions of the belonging function for Q_{ov}, NQ_{ov}, and Q_{ds}^{ov}. Therefore, we have $Q_{ov}(average)$ with $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c} + a_{j,c}}{2}$, $Q_{ov}(product)$ with $f(a_{i,c}, a_{j,c}) = a_{i,c} a_{j,c}$, Q_{ov}^{L} in Equation (4.13), $NQ_{ov}(average)$ with $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c} + a_{j,c}}{2}$, $NQ_{ov}(product)$ with $f(a_{i,c}, a_{j,c}) = a_{i,c} a_{j,c}$, $Q_{ds}^{ov}(average)$ with $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c} + a_{j,c}}{2}$, and $Q_{ds}^{ov}(product)$ with $f(a_{i,c}, a_{j,c}) = a_{i,c} a_{j,c}$. For fuzzy overlapping community structure, $a_{i,c}$ is given

for each node i to each community c to which it belongs. For crisp overlapping community structures, we can adopt Equation (4.5) and Equation (4.10) to calculate $a_{i,c}$. Consequently, two versions of the belonging coefficient can be used to convert crisp overlapping to fuzzy overlapping.

We also consider eight local community quality metrics: the number of *Intra-edges*, *Intra-density*, *Contraction*, the number of *Boundary-edges*, *Expansion*, *Conductance* [21, 22, 69], the *Fitness* function [70], and the *Average Modularity Degree* [71]. These metrics describe how the connectivity structure of a given set of nodes resembles a community. All of them rely on the intuition that communities are sets of nodes with many edges inside them and few edges outside of them. We also extend these metrics to be applicable to fuzzy overlapping community structure in which nodes are assigned probability of belonging to each community of which they are part. Two versions of the belonging coefficient and two versions of the belonging function are considered for each metric. For fuzzy overlapping community structure, we define the size of a community c as $|c| = \sum_{i \in c} a_{i,c}$.

The number of *Intra-edges* (*IE*): $|E_c^{in}|$; it is the total number of edges in c. A large value of this metric is better than a small value in terms of the community quality.

***Intra-density* (*ID*):** d_c in Equation (4.19). The larger the value of this metric, the higher the quality of the communities.

***Contraction* (*CNT*):** $2|E_c^{in}|/|c|$; it measures the average number of edges per node inside the community c. A larger value of contraction means a better community quality.

The number of *Boundary-edges* (BE): $|E_c^{out}|$; it is the total number of edges on the boundary of c. A small value of this metric is better than a large value in terms of the community quality.

***Expansion* (*EXP*):** $|E_c^{out}|/|c|$; it measures the average number of edges (per node) that point outside the community c. A smaller value of expansion corresponds to a better community structure.

***Conductance* (*CND*):** $\frac{|E_c^{out}|}{2|E_c^{in}|+|E_c^{out}|}$; it measures the fraction of the total number of edges that point outside the community. A smaller value of conductance means

a better community quality.

The *Fitness* (F) function: $\frac{|E_c^{in}|}{|E_c^{in}|+|E_c^{out}|}$; it is the ratio between the internal degree and the total degree of a community c. A larger value of F indicates a better community quality.

***Average Modularity Degree* (D):** $\sum_{c \in C} \frac{2|E_c^{in}|-|E_c^{out}|}{|c|}$; it is the summation of the average modularity degree of each community. The average modularity degree of a community ($\frac{2|E_c^{in}|-|E_c^{out}|}{|c|}$) equals to the average inner degree ($\frac{2|E_c^{in}|}{|c|}$) minus the average outer degree ($\frac{|E_c^{out}|}{|c|}$). The larger the value of D, the higher the quality of the community structure.

In this section, we compare different choices of the belonging coefficient and the belonging function to be used for Q_{ov}, Q_{ov}^{L}, NQ_{ov}, Q_{ds}^{ov}, and the eight local community quality metrics in order to see which version of the belonging coefficient and which version of the belonging function are better. Then, we try to determine which of the three overlapping extensions of modularity (two kinds of node-based extensions of modularity and the edge-based extension of modularity) is the best. In addition, we compare the performance of all these overlapping metrics with the best combination of belonging coefficient and belonging function to recommend which metrics to select when measuring the quality of overlapping community structure.

The experiments are done with three community detection algorithms, Speaker-listener Label Propagation Algorithm (SLPA) [72, 73], Clique Percolation Method (CFinder) [23], and SpeakEasy [74] which is a label propagation algorithm specialized for biology networks, on a large number of real-world networks and synthetic networks. We vary the threshold parameter r of SLPA [72,73] from 0.05 to 0.5 with step 0.05. SLPA gets crisp overlapping communities when $r < 0.5$ and gets disjoint communities when $r = 0.5$ (for $r > 0.5$ SLPA generates the same disjoint communities as for $r = 0.5$). For each value of threshold r, we adopt 10 running samples since the community detection result of SLPA is not deterministic. We vary the parameter k of CFinder from 3 to 20 with step 1 but only when such k-clique-community is available. It is usually the case that some nodes are not in the final discovered k-clique-communities so we consider each of these nodes forming a community of itself. The threshold parameter tr of SpeakEasy is varied from 0.05 to 1 with step

0.05. SpeakEasy gets crisp overlapping community structure when $tr < 1$ and gets disjoint community structure when $tr = 1$.

Then, for the community detection results of SLPA with different values of threshold r, the results of CFinder with different values of k, and the results of SpeakEasy with different thresholds tr on each of these networks, we calculate the values of Q_{ov}, Q^L_{ov}, NQ_{ov} Q^{ov}_{ds}, and the eight local community quality metrics (twelve metrics in total) with two versions of the **belonging coefficient (BC)** and two versions of the **belonging function (BF)**. For each r of SLPA, the values of all the metrics are calculated as the average of the 10 runs. For convenience, we denote Equation (4.5) and Equation (4.10) as the first and the second version of the belonging coefficient, respectively. We also denote the belonging function $f(a_{i,c}, a_{j,c}) = \frac{a_{i,c}+a_{j,c}}{2}$ as the first version of the belonging function and $f(a_{i,c}, a_{j,c}) = a_{i,c}a_{j,c}$ as the second version of the belonging function. We determine which version of the belonging coefficient and which version of the belonging function are better based on the largest number of quality metrics consistent with each other on determining the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy on all these networks. Finally, we compare the performance of these overlapping metrics with the best combination of belonging coefficient and belonging function by looking at how many times each metric is among those that are consistent with each other on determining the best values of parameters for the three adopted community detection algorithms.

4.4.1 Real-world Network Datasets

We consider totally 23 real-world network datasets, including friendship network, collaboration networks, co-purchasing networks, biology networks, etc. Table 4.1 shows the basic properties of all these datasets. It can be seen that these networks have different numbers of nodes and different numbers of edges, varying from very small networks to very large networks. Moreover, edges in some networks have weights and directions.

Celegans is a metabolic network of C. elegans [45]. **Dolphin** is an social network of frequent associations between 62 dolphins in a community living off

Table 4.1: Basic properties of all real-world network datasets used in the experiments.

Name	#Nodes	#Edges	Type	Description
Celegans	453	2025	Unweighted & Undirected	Metabolic network of C. elegans [45]
Dolphin	62	159	Unweighted & Undirected	Dolphin social network [76]
Email	1133	5451	Unweighted & Undirected	Email network [77]
Football	115	613	Unweighted & Undirected	American college football network [78]
Jazz	198	2742	Unweighted & Undirected	Jazz musicians network [79]
Karate	34	78	Unweighted & Undirected	Zachary's karate club network [80]
Lesmis	77	254	Weighted & Undirected	Characters coappearance network [81]
Netscience	1461	2742	Weighted & Undirected	Coauthorship network [39]
PGP	10680	24316	Unweighted & Undirected	PGP network [82]
Polblogs	1224	19022	Unweighted & Directed	Political blogs network [83]
Polbooks	105	441	Unweighted & Undirected	Network of books about US politics [84]
Railway	297	1213	Unweighted & Undirected	Indian railway network [85]
Santafe	118	200	Unweighted & Undirected	Collaboration network of scientists [78]
Collins_cyc	1097	6392	Unweighted & Undirected	Protein-protein interaction networks [74]
Collins_cyc_w	1097	6392	Weighted & Undirected	
Collins_mips	734	4778	Unweighted & Undirected	
Collins_sgd	809	2955	Unweighted & Undirected	
Gavin_cyc	997	4031	Unweighted & Undirected	
Gavin_cyc_w	997	4031	Weighted & Undirected	
Gavin_mips	701	2695	Unweighted & Undirected	
Gavin_sgd	747	2639	Unweighted & Undirected	
Amazon	319948	880215	Unweighted & Undirected	Amazon product network [33]
DBLP	260998	950059	Unweighted & Undirected	DBLP collaboration network [33]

Doubtful Sound, New Zealand [76]. **Email** is a network of email interchanges between members of the Univeristy Rovira i Virgili (Tarragona) [77]. **Football** is a network that represents the schedule of games between college football teams in a single season [78]. **Jazz** is a network of collaborations between jazz musicians [79]. **Karate** is a network representing the friendships between 34 members of a karate club at a US university during two years [80]. **Lesmis** is a coappearance network of characters in the novel Les Miserables [81]. **Netscience** is a coauthorship network of scientists working on network theory and experiment [39]. **PGP** is the giant component of the network of users of the Pretty-Good-Privacy (PGP) algorithm for secure information interchange [82]. **Polblogs** is a directed network of hyperlinks between weblogs on US politics, recorded in 2005 by Adamic and Glance [83]. **Polbooks** is a network of books about US politics published around the time of the 2004 presidential election and sold by the online bookseller Amazon.com [84]. **Railway** is a network with nodes representing Indian railway stations, where t-

wo stations are connected by an edge if there exists at least one train-route such that both stations are scheduled stops on that route [85]. **Santafe** is the largest connected component of the collaboration network of scientists at the Santa Fe Institute during years 1999 and 2000 [78]. **Collins_cyc**, **Collins_cyc_w**, **Collins_mips**, **Collins_sgd**, **Gavin_cyc**, **Gavin_cyc_w**, **Gavin_mips**, and **Gavin_sgd** are two kinds (referred as Collins [86] and Gavin [87] here) of popular high throughput protein-protein interaction networks derived from measurements obtained by affinity purification and mass spectrometry (AP-MS) techniques [74]. These two kinds of networks are further refined with three gold-standards for protein complexes, including the classic Munich Information Center for Protein Sequences (MIPS) [88] and the more recent Saccharomyces Genome Database (SGD) [89]. The complete MIPS dataset as well as partial information from SGD are incorporated into a third protein complex list known as CYC2008 [90]. Thus, we have **Collins_cyc**, **Collins_mips**, **Collins_sgd**, **Gavin_cyc**, **Gavin_mips**, and **Gavin_sgd**, respectively. **Collins_cyc_w** and **Gavin_cyc_w** are respectively the weighted versions of **Collins_cyc** and **Gavin_cyc**, in which the weight is proportional to the probability a given interaction pair truly exists. **Amazon** is a product co-purchasing network of the Amazon website [33]. The nodes of the network represent products and edges link commonly copurchased products. **DBLP** is a scientific collaboration network where nodes represent authors and edges connect authors that have co-authored a paper [33].

Tables 4.2-4.4 show the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy, respectively, along with the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best r, this best k, and this best tr for the four combinations of two versions of belonging coefficient and two versions of belonging function on all 23 real-world network datasets. For instance, entry **0.5 (5)** in row **Dolphin** and column **(BC,BF)=(1,1)** in Table 4.2 means that there are totally five out of twelve community quality metrics showing that the best value of threshold r for SLPA is 0.5 when adopting the first version of belonging coefficient and the first version of belonging function. The red italic font

Table 4.2: The best value of threshold r for SLPA and the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best r for the four combinations of two versions of belonging coefficient and two versions of belonging function on all real-world network datasets. The best value in each row is marked by red italic font.

	SLPA			
Datasets	(BC,BF)=(1,1)	(BC,BF)=(1,2)	(BC,BF)=(2,1)	(BC,BF)=(2,2)
Celegans	***0.5 (6)***	0.05 (4)	***0.5 (6)***	0.4 (5)
Dolphin	0.5 (5)	***0.4 (8)***	0.5 (6)	{0.05,0.4} (5)
Email	0.5 (8)	***0.5 (10)***	0.5 (7)	0.5 (8)
Football	{0.45,0.5} (9)	***{0.45,0.5} (11)***	{0.45,0.5} (8)	0.25 (7)
Jazz	***0.5 (10)***	***0.5 (10)***	0.5 (9)	0.5 (9)
Karate	0.5 (8)	***0.45 (10)***	0.5 (7)	***0.45 (10)***
Lesmis	{0.25,0.5} (4)	***0.25 (6)***	0.5 (4)	0.15 (5)
Netscience	0.35 (4)	0.5 (4)	***0.35 (5)***	{0.35,0.5} (3)
PGP	0.5 (8)	***0.5 (10)***	0.5 (7)	0.5 (8)
Polblogs	0.5 (7)	***0.5 (8)***	0.5 (6)	0.5 (4)
Polbooks	***0.5 (7)***	0.2 (5)	***0.5 (7)***	0.2 (4)
Railway	0.5 (8)	***0.5 (9)***	0.5 (7)	0.5 (7)
Santafe	0.4 (6)	***0.4 (7)***	0.4 (5)	0.4 (5)
Collins_cyc	***0.5 (8)***	0.5 (5)	***0.5 (8)***	0.5 (6)
Collins_cyc_w	0.05 (7)	0.05 (7)	0.05 (6)	***0.05 (8)***
Collins_mips	0.45 (5)	***0.45 (6)***	{0.05,0.45} 4	{0.05,0.45} (4)
Collins_sgd	0.5 (4)	0.1 (4)	0.5 (4)	***0.1 (6)***
Gavin_cyc	0.45 (5)	0.45 (6)	0.45 (5)	***0.45 (7)***
Gavin_cyc_w	0.35 (5)	***0.3 (6)***	0.05 (4)	0.3 (5)
Gavin_mips	0.5 (9)	***0.5 (11)***	0.5 (8)	0.5 (6)
Gavin_sgd	0.5 (8)	***0.5 (9)***	0.5 (7)	0.5 (7)
Amazon	0.5 (8)	***0.5 (10)***	0.5 (6)	0.5 (9)
DBLP	0.5 (7)	***0.5 (10)***	0.5 (6)	0.5 (8)

in each dataset row of these tables denotes the best combination of the two versions of belonging coefficient and two versions of belonging function for each dataset. For example, ***0.4 (8)*** in row **Dolphin** and column **(BC,BF)=(1,2)** in Table 4.2 indicates that the twelve metrics with the first version of belonging coefficient and the second version of belonging function is the best since the number of metrics (which is eight here) consistent with each other on determining the best value of r is the largest among the four combinations.

It can be observed from Table 4.2 that almost all the networks, except **Cel-**

Table 4.3: The best value of parameter k for CFinder and the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best k for the four combinations of two versions of belonging coefficient and two versions of belonging function on all real-world network datasets. The best value in each row is marked by red italic font.

	CFinder			
Datasets	(BC,BF)=(1,1)	(BC,BF)=(1,2)	(BC,BF)=(2,1)	(BC,BF)=(2,2)
Celegans	3 (6)	***3 (7)***	3 (5)	{3,9} (4)
Dolphin	3 (11)	***3 (12)***	3 (7)	3 (7)
Email	***3 (10)***	***3 (10)***	4 (4)	{3,4,9-12} (3)
Football	***4 (6)***	{3,4} (5)	4 (5)	4 (5)
Jazz	***3 (8)***	***3 (8)***	3 (6)	3 (5)
Karate	***3 (11)***	***3 (11)***	3 (8)	3 (7)
Lesmis	3 (7)	***3 (8)***	6 (4)	{3,6} (4)
Netscience	3 (11)	***3 (12)***	3 (9)	3 (9)
PGP	3 (10)	***3 (12)***	3 (6)	3 (8)
Polblogs	N/A	N/A	N/A	N/A
Polbooks	***3 (10)***	***3 (10)***	3 (7)	3 (6)
Railway	***3 (9)***	***3 (9)***	3 (4)	3 (4)
Santafe	3 (10)	***3 (11)***	3 (7)	3 (8)
Collins_cyc	***3 (11)***	***3 (11)***	3 (7)	3 (8)
Collins_cyc_w	N/A	N/A	N/A	N/A
Collins_mips	***3 (10)***	***3 (10)***	3 (6)	3 (6)
Collins_sgd	***3 (12)***	***3 (12)***	3 (8)	3 (8)
Gavin_cyc	3 (11)	***3 (12)***	3 (6)	3 (6)
Gavin_cyc_w	N/A	N/A	N/A	N/A
Gavin_mips	***3 (12)***	***3 (12)***	3 (7)	3 (6)
Gavin_sgd	***3 (11)***	***3 (11)***	3 (6)	3 (6)
Amazon	***3 (11)***	***3 (11)***	3 (6)	3 (6)
DBLP	3 (9)	***3 (10)***	3 (5)	3 (5)

egans, **Netscience**, **Polbooks**, **Collins_cyc**, **Collins_cyc_w**, **Collins_sgd**, and **Gavin_cyc**, imply that (BC,BF)=(1,2) is the best among the four possible combinations when using SLPA. Table 4.3 shows that (BC,BF)=(1,2) is the best on all the networks, except **Football**, when using CFinder. Results for CFinder on **Polblogs**, **Collins_cyc_w**, and **Gavin_cyc_w** are not provided because it has not finished running on these three networks for more than two months processing many potential k-cliques. Similarly, Table 4.4 indicates that (BC,BF)=(1,2) is the best on all the networks, except **Jazz**, when using SpeakEasy. We can observe from the

Table 4.4: The best value of threshold tr for SpeakEasy and the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best tr for the four combinations of two versions of belonging coefficient and two versions of belonging function on all real-world network datasets. The best value in each row is marked by red italic font.

	SpeakEasy			
Datasets	(BC,BF)=(1,1)	(BC,BF)=(1,2)	(BC,BF)=(2,1)	(BC,BF)=(2,2)
Celegans	0.75 (6)	***0.75 (7)***	0.75 (6)	0.75 (6)
Dolphin	0.4 (3)	***0.4 (4)***	***0.15 (4)***	***0.7 (4)***
Email	***0.9 (3)***	***1 (3)***	{0.05,0.9} (2)	***{0.5,1} (3)***
Football	***0.6 (10)***	***0.6 (10)***	***0.6 (10)***	***0.6 (10)***
Jazz	0.75 (5)	0.75 (5)	0.75 (5)	***0.75 (6)***
Karate	0.45 (5)	***0.45 (6)***	0.45 (5)	***0.45 (6)***
Lesmis	0.85 (5)	***0.85 (6)***	0.85 (4)	0.85 (4)
Netscience	0.7 (4)	***0.25 (10)***	0.05 (3)	{0.15,0.2,0.25,0.7} (2)
PGP	0.85 (5)	***0.85 (7)***	{0.05,0.75,0.85} (3)	0.85 (4)
Polblogs	***0.7 (4)***	***0.7 (4)***	0.45 (3)	{0.5,0.7,0.8,0.9} (2)
Polbooks	0.95 (5)	***0.95 (6)***	{0.5,0.95} (3)	0.95 (4)
Railway	0.8 (4)	***0.8 (5)***	0.8 (3)	0.8 (4)
Santafe	***0.9 (5)***	***0.9 (5)***	0.9 (4)	{0.65,0.9} (4)
Collins_cyc	0.9 (8)	***0.9 (10)***	0.9 (4)	0.9 (6)
Collins_cyc_w	0.55 (6)	***0.55 (9)***	{0.1,0.55} (3)	0.55 (4)
Collins_mips	0.4 (7)	***0.4 (10)***	0.4 (5)	{0.25,0.4,0.6} (3)
Collins_sgd	0.8 (7)	***0.8 (10)***	{0.5,0.8} (4)	{0.5,0.8} (5)
Gavin_cyc	0.7 (6)	***0.7 (8)***	0.7 (5)	0.7 (6)
Gavin_cyc_w	0.7 (4)	***0.7 (5)***	{0.05,0.5,0.7,0.95} (2)	0.7 (3)
Gavin_mips	0.9 (6)	***0.9 (8)***	0.9 (5)	0.9 (5)
Gavin_sgd	0.8 (5)	***0.8 (6)***	***0.8 (6)***	***0.8 (6)***
Amazon	1 (9)	***1 (11)***	1 (8)	***1 (11)***
DBLP	1 (9)	***1 (10)***	1 (8)	***1 (10)***

three tables that for each network there are at least two out of three algorithms (SLPA, CFinder, and SpeakEasy) supporting conclusion that (BC,BF)=(1,2) is the best. Thus, we determined that the first version of the belonging coefficient is better than the alternative. It means that to convert crisp overlapping to fuzzy overlapping the belonging coefficient of a node to a community should be the reciprocal of the number of communities of which this node is a part. When the relationship between a node and the communities to which it belongs is binary, there is no information about the strength of the membership. In this case, it is intuitive and

reasonable to assign a node to its communities using equal belonging coefficients. We also determined that the second version of the belonging function is better than the alternative. It means that the probability of the event that two nodes belong to the same community should be the product, not the average, of their belonging coefficients to that community. In addition, $Q_{ov} = Q'_{ov}$ when $f(a_{i,c}, a_{j,c}) = a_{i,c}a_{j,c}$ as proved in Subsection 4.1, which is another way of showing that the second version of the belonging function is much more suitable for use in the metric than the first. Therefore, we conclude that the overlapping community quality metrics with the first version of the belonging coefficient and the second version of the belonging function are the best among the four possible combinations on all the real-world network datasets. Tables 1-23 in APPENDIX A.2 contain more detailed results on these real-world network datasets.

Table 4.5 shows the number of times (maximum three since there are totally three community detection algorithms adopted) that the community quality metric is among the quality metrics that are consistent with each other on determining the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy on each real-world network with (BC,BF)=(1,2). The last row in the table presents on how many networks this metric got support from all three algorithms and how many times on average (maximum three; but for **Polblogs**, **Collins_cyc_w**, and **Gavin_cyc_w** the maximum value is two because CFinder has no result for these networks) this metric got support from each of the 23 real-world networks. This table indicates that the edge-based overlapping definition is the best overlapping extension for modularity among the three extensions (two kinds of node-based extensions of modularity and the edge-based extension of modularity). The fitness function gets the largest values in the last row and generally local metrics are better than global metrics here which may imply that local community quality metrics are more applicable for measuring the quality of overlapping community structure. Note that eight out of twelve metrics are local metrics, so they are majority here which may impact the selection of the most consistent metrics. In addition, even though we have already shown in [21, 22, 91] that modularity density simultaneously resolves two opposite yet coexisting

Table 4.5: The number of times (maximum three since there are totally three community detection algorithms adopted) that the community quality metric is among those consistent with each other on determining the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy on each real-world network with (BC,BF)=(1,2). The best value in the last row is marked by red italic font.

Datasets	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D
Celegans	1	1	1	1	3	0	2	0	2	2	3	2
Dolphin	3	2	3	2	2	2	1	1	2	2	1	3
Email	2	2	3	2	3	1	1	1	2	2	2	2
Football	2	1	3	3	3	2	3	2	3	3	3	3
Jazz	2	2	2	2	2	1	1	3	2	2	2	2
Karate	3	3	3	3	2	1	2	1	3	2	2	2
Lesmis	0	2	1	0	2	2	3	0	2	3	3	2
Netscience	3	2	3	2	3	1	1	3	2	2	2	2
PGP	3	2	3	2	2	1	3	2	3	3	2	3
Polblogs	1	1	1	1	1	0	1	1	1	1	1	2
Polbooks	2	1	2	1	2	2	1	0	2	3	3	2
Railway	1	1	1	1	2	1	3	2	2	3	3	3
Santafe	2	2	2	1	1	2	3	0	2	3	3	2
Collins_cyc	3	2	3	1	2	3	2	2	2	2	2	2
Collins_cyc_w	2	1	2	0	2	0	1	1	2	2	2	1
Collins_mips	2	3	3	1	2	2	2	1	3	2	2	3
Collins_sgd	2	2	2	1	3	2	3	1	2	3	3	2
Gavin_cyc	2	2	2	2	2	1	1	2	3	3	3	3
Gavin_cyc_w	0	0	0	1	1	0	2	0	1	2	2	2
Gavin_mips	3	2	3	2	2	2	3	2	3	3	3	3
Gavin_sgd	2	3	2	1	2	2	3	1	2	3	3	2
Amazon	3	3	3	3	3	2	3	1	3	2	3	3
DBLP	2	2	3	3	3	2	2	1	3	3	3	3
	8 (2)	4 (1.83)	12 (2.22)	4 (1.57)	8 (2.17)	1 (1.39)	10 (2.04)	2 (1.22)	9 (2.26)	13 (*2.43*)	*14* (*2.43*)	11 (2.35)

problems of modularity, Q^{ov}_{ds} is not highly consistent with majority of other metrics. This might be because only one other metric consider community density. We will investigate if adding more community density based metrics can change the above results.

4.4.2 LFR Benchmark Networks

LFR (named after the initials of names of authors) benchmark networks [92] have become a standard in the evaluation of the performance of community detection algorithms. The LFR benchmark network that we used here has 1000 nodes with average degree 15 and maximum degree 50. The exponent γ for the degree sequence varies from 2 to 3. The exponent β for the community size distribution ranges from 1 to 2. Then, four pairs of the exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1), \text{and } (3, 2)$ are chosen in order to explore the widest spectrum of graph structures. The mixing parameter μ is varied from 0.05 to 0.95. It means that each node shares a fraction $(1-\mu)$ of its edges with the other nodes in its community and shares a fraction μ of its edges with the nodes outside its community. Thus, low mixing parameters indicate strong community structure. The degree of overlap is determined by two parameters. O_n is the number of overlapping nodes, and O_m is the number of communities to which each overlapping node belongs. O_n here is set to 10% of the total number of nodes. Instead of fixing O_m, we allow it to vary from 1 to 8 indicating the overlapping diversity of overlapping nodes. By increasing the value of O_m, we create harder detection tasks. Also, we generate 10 network instances for each configuration of these parameters. Hence, each metric value for a certain configuration of LFR represents the average metric values of all 10 instances. Since the experimental results are similar for all four pairs of exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1), \text{and } (3, 2)$, for the sake of brevity, we only present the results for $(\gamma, \beta) = (2, 1)$ here. In addition, these results are similar for different values of μ and we cannot show all the results in one page as can be observed from Tables 24-26 in APPENDIX A.3, so here we only show the results for $\mu = 0.3, 0.35, \text{and } 0.4$. We choose $\mu = 0.3, 0.35, \text{and } 0.4$ to better illustrate the results since with $\mu = 0.3, 0.35, \text{and } 0.4$ the community structures generated by LFR are around the boundary of well-separated communities and well-connected communities. For each node, $\mu = 0.5$ means that the number of its edges with other nodes in its communities is equal to the number of its edges with nodes outside its community, which makes the community structure difficult to discover.

Tables 4.6-4.8 show the best value of threshold r for SLPA, the best value of

Table 4.6: The best value of threshold r for SLPA and the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best r for the four combinations of two versions of belonging coefficient and two versions of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. The best value in each row is marked by red italic font.

		SLPA			
μ	O_m	(BC,BF)=(1,1)	(BC,BF)=(1,2)	(BC,BF)=(2,1)	(BC,BF)=(2,2)
0.3	1	0.5 (7)	***0.5 (8)***	0.5 (7)	0.5 (6)
	2	***0.5 (8)***	0.5 (5)	0.5 (7)	0.3 (4)
	4	0.5 (9)	***0.5 (10)***	0.5 (8)	0.5 (5)
	6	0.5 (9)	***0.5 (10)***	0.5 (8)	0.5 (6)
	8	0.5 (9)	***0.5 (10)***	0.5 (8)	0.5 (9)
0.35	1	0.25 (5)	***0.25 (6)***	0.25 (4)	0.25 (5)
	2	***0.5 (7)***	***0.45 (7)***	***0.5 (7)***	0.45 (6)
	4	0.5 (8)	***0.5 (9)***	0.5 (7)	0.5 (6)
	6	0.5 (8)	***0.5 (9)***	0.5 (7)	0.5 (8)
	8	0.5 (7)	***0.5 (9)***	0.5 (6)	0.5 (8)
0.4	1	***0.5 (6)***	0.25 (5)	0.5 (4)	{0.2,0.25,0.45} (3)
	2	0.5 (8)	***0.5 (9)***	0.5 (7)	0.5 (5)
	4	0.5 (9)	***0.5 (10)***	0.5 (8)	0.5 (8)
	6	0.5 (9)	***0.5 (10)***	0.5 (9)	0.5 (9)
	8	0.5 (8)	***0.5 (9)***	0.5 (7)	0.5 (8)

parameter k for CFinder, and the best value of threshold tr for SpeakEasy, respectively, along with the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best r, this best k, and this best tr for the four possible combinations of two versions of belonging coefficient and two versions of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. Table 4.6 implies that (BC,BF)=(1,2) is the best among the four possible combinations on all configurations of LFR networks except $\mu = 0.3, O_m = 2$ and $\mu = 0.4, O_m = 1$ when using SLPA. Table 4.7 demonstrates that (BC,BF)=(1,2) is the best on all configurations of LFR networks when using CFinder. Table 4.8 indicates that (BC,BF)=(1,2) is the best among the four combinations on all configurations of LFR networks except $\mu = 0.35, O_m = 6$ and $\mu = 0.4, O_m = 2, 4$ when using SpeakEasy. Consequently, we could conclude that the overlapping community quality metrics with the first version of belonging coefficient and the second version of the belonging function are the best among the four possi-

Table 4.7: The best value of parameter k for CFinder and the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best k for the four combinations of two versions of belonging coefficient and two versions of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. The best value in each row is marked by red italic font.

		CFinder			
μ	O_m	(BC,BF)=(1,1)	(BC,BF)=(1,2)	(BC,BF)=(2,1)	(BC,BF)=(2,2)
0.3	1	***4 (9)***	***4 (9)***	4 (7)	4 (6)
	2	{3,4} (5)	***4 (6)***	3 (5)	4 (5)
	4	***3 (6)***	***3 (6)***	3 (4)	3 (4)
	6	3 (5)	***4 (7)***	{3,4,11} (3)	4 (4)
	8	3 (5)	***4 (7)***	{3,4,12} (3)	4 (4)
0.35	1	***4 (8)***	***4 (8)***	4 (6)	4 (5)
	2	***3 (6)***	***3 (6)***	{3,8} (4)	4 (4)
	4	***3 (6)***	***4 (6)***	3 (4)	4 (4)
	6	4 (5)	***4 (7)***	4 (5)	4 (5)
	8	4 (6)	***4 (7)***	4 (4)	4 (5)
0.4	1	***4 (6)***	***3 (6)***	{3,4,8} (4)	4 (4)
	2	***3 (6)***	***3 (6)***	{3,9} (4)	4 (4)
	4	***3 (6)***	***4 (6)***	{3,9} (4)	4 (4)
	6	4 (7)	***4 (8)***	4 (5)	4 (5)
	8	4 (6)	***4 (7)***	4 (5)	4 (5)

ble combinations on LFR networks. Please refer to Tables 24-26 in APPENDIX A.3 for more detailed results on LFR benchmark networks.

Table 4.9 shows the number of times (maximum three since there are totally three community detection algorithms adopted) that the community quality metric is among the metrics that are consistent with each other on determining the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy on each configuration of LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$ when (BC,BF)=(1,2). The last row in the table presents how many different configurations of LFR networks support this metric across all three algorithms and how many times on average (maximum three) this metric got support from each of the 15 different configurations. This table shows results similar with those presented in Table 4.5. Node-based overlapping extension and edge-based overlapping extension of modularity perform equally well on LFR networks. The last row shows that the fitness function has the largest values

Table 4.8: The best value of threshold tr for SpeakEasy and the corresponding number of quality metrics (out of twelve) that are consistent with each other on determining this best tr for the four combinations of two versions of belonging coefficient and two versions of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. The best value in each row is marked by red italic font.

		SpeakEasy			
μ	O_m	(BC,BF)=(1,1)	(BC,BF)=(1,2)	(BC,BF)=(2,1)	(BC,BF)=(2,2)
0.3	1	***0.75 (9)***	***0.75 (9)***	***0.75 (9)***	***0.75 (9)***
	2	0.8 (7)	***0.8 (8)***	0.8 (6)	0.8 (7)
	4	***1 (4)***	***1 (4)***	***1 (4)***	***1 (4)***
	6	***0.95 (4)***	***0.95 (4)***	0.05 (3)	***{0.35,0.7} (4)***
	8	0.05 (3)	***{0.4,0.6} (4)***	{0.05,0.85} (3)	{0.4,0.85} (3)
0.35	1	***0.8 (9)***	***0.8 (9)***	***0.8 (9)***	***0.8 (9)***
	2	0.95 (5)	***0.95 (6)***	0.95 (4)	{0.35,0.95} (3)
	4	{0.05,0.95} (3)	***0.75 (4)***	{0.05,0.95} (3)	0.95 (3)
	6	***0.85 (5)***	0.85 (4)	0.85 (4)	0.45 (3)
	8	{0.05,0.85} (3)	***0.9 (5)***	{0.05,0.85,1} (3)	{0.85,0.9} (3)
0.4	1	***0.15 (9)***	***0.15 (9)***	***0.15 (9)***	***0.15 (9)***
	2	{0.8,0.85,0.95} (3)	0.95 (4)	{0.05,0.85,0.95} (3)	***0.95 (5)***
	4	0.95 (5)	***0.95 (6)***	0.95 (4)	0.95 (4)
	6	0.95 (4)	{0.65,0.75} (3)	0.75 (4)	***0.75 (6)***
	8	{0.05,0.9,1} (3)	***0.4 (5)***	{0.05,0.95,1} (3)	{0.4,0.5} (3)

and generally local metrics are better than global metrics, and still Q_{ds}^{ov} performs poor. The reason is the same as given for the results in Table 4.5.

4.5 Summary

In this chapter, we determined which versions of the belonging coefficient and the belonging function are better for measuring quality of fuzzy overlapping community structure. We found that the first version of the belonging coefficient is better than the second one, which means that the coefficient of a node belonging to a community should be the reciprocal of the number of communities to which this node belongs. In addition, we found that the second version of the belonging function is better than the first version, meaning that the probability that two nodes belong to the same community should be the product, not the average, of their belonging coefficients. Moreover, we proposed overlapping extensions for localized modularity, modularity density, and eight local community quality metrics analogous to such

Table 4.9: The number of times (maximum three since there are totally three community detection algorithms adopted) that the community quality metric is among those consistent with each other on determining the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy on each configuration of LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$ when (BC,BF)=(1,2). The best value in the last row is marked by red italic font.

μ	O_m	Q_{ov}	NQ_{ov}	Q_{ov}^{L}	Q_{ds}^{ov}	IE	ID	CNT	BE	EXP	CND	F	D
0.3	1	3	2	3	3	1	0	2	0	3	3	3	3
	2	3	1	3	2	1	1	2	0	2	1	1	2
	4	1	2	1	1	1	3	2	2	1	2	2	2
	6	2	1	2	2	0	1	2	0	3	3	3	2
	8	3	1	3	2	1	1	3	0	3	3	3	2
0.35	1	3	1	3	2	1	0	2	0	3	2	3	3
	2	2	0	2	1	1	2	2	1	1	2	2	3
	4	2	2	2	2	0	0	3	0	2	2	2	2
	6	2	1	2	2	0	0	2	0	3	3	3	2
	8	3	1	3	2	1	0	2	0	3	2	2	2
0.4	1	1	0	1	1	2	1	3	0	2	3	3	3
	2	2	1	2	1	1	2	2	0	1	2	2	3
	4	3	2	3	2	0	1	3	0	2	2	2	2
	6	3	2	3	3	0	1	3	0	2	3	3	1
	8	2	1	2	2	1	0	3	0	2	3	3	2
		7 (2.33)	0 (1.2)	7 (2.33)	2 (1.87)	0 (0.73)	1 (0.87)	6 (2.4)	0 (0.2)	6 (2.2)	7 (2.4)	***8*** ***(2.47)***	5 (2.27)

extension of modularity. Based on the experimental results, we recommend using the edge-based overlapping extension of modularity with the first version of belonging coefficient and with its own belonging function. We also recommend using the node-based overlapping extension of modularity and overlapping extension of modularity density with the first version of belonging coefficient and the second version of belonging function as the metrics of the global quality of overlapping community structure.

In the future, we plan to explore local community quality metrics for overlapping community structure and investigate more community quality metrics incorporating community density to see whether putting community density into these metrics will make them perform better.

CHAPTER 5
FINE-TUNED DISJOINT COMMUNITY DETECTION ALGORITHMS

In this chapter, we propose a novel fine-tuned disjoint community detection algorithm that repeatedly attempts to improve the quality measures by splitting and merging the given community structure. We denote the corresponding algorithm based on modularity (Q) as *Fine-tuned* Q while the one based on modularity density (Q_{ds}) is referred to as *Fine-tuned* Q_{ds}. Finally, we evaluate the greedy algorithm of modularity maximization (denoted as *Greedy* Q), *Fine-tuned* Q, and *Fine-tuned* Q_{ds} by using seven community quality metrics based on ground truth communities. These evaluations are conducted on four real-world networks, and also on the classical clique network and the LFR benchmark networks, each of which is instantiated by a wide range of parameters. The results indicate that *Fine-tuned* Q_{ds} is the most effective method and can also dramatically improve the community detection results of other algorithms. Further, all seven quality metrics based on ground truth communities are consistent with Q_{ds}, but not consistent with Q, which implies the superiority of modularity density over the original modularity.

5.1 Maximizing Modularity Density (Q_{ds})

In Chapter 3, we have given the definition of modularity density (Q_{ds}). With formal proofs and experiments on two real-world dynamic datasets (Senate dataset [66] and Reality Mining Bluetooth Scan data [67]) we demonstrated that Q_{ds} eliminates the two opposite yet coexisting problems of modularity: the problem of favoring small communities and the problem of favoring large communities (also called the resolution limit problem). Moreover, for a given community in Q_{ds} defined by Equation (3.6), its internal and pair-wise densities and its split penalty are local components, which is related to the *resolution-limit-free* definition in [61].

Portions of this chapter previously appeared as: M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE T. Comput. Soc. Syst.*, vol. 1, no. 1, pp. 46-65, Mar. 2014.

Therefore, it is reasonable to expect that maximizing Q_{ds} would discover more meaningful community structure than maximizing Q. In this section, we first illustrate why the greedy agglomerative algorithm for increasing Q_{ds} cannot be adopted for optimizing Q_{ds}. Then, we propose a fine-tuned community detection algorithm that repeatedly attempts to improve the community quality measurements by splitting and merging the given network community structure to maximize Q_{ds}.

5.1.1 Greedy Algorithm Fails to Optimize Q_{ds}

In this subsection, we show why the greedy agglomerative algorithm increasing Q_{ds} fails to optimize it. At the first step of the greedy algorithm for increasing Q_{ds}, each node is treated as a single community. Then, Q_{ds} of each node or community is $Q_{ds} = -SP$. Therefore, in order to increase Q_{ds} the most, the greedy algorithm would first merge the connected pair of nodes with the sum of their degrees being the largest among all connected pairs. However, it is very likely that those two nodes belong to two different communities, which would finally result in merging those two communities instead of keeping them separate. This will result in a much lower value of Q_{ds} for such a merged community compared to Q_{ds} for its components, demonstrating the reason for greedy Q_{ds} algorithm failure in optimizing Q_{ds}.

For example, in the network example in Figure 5.1, the initial values of Q_{ds} for nodes $1, 2, 4, 6, 7,$ and 8 with degree 3 are $Q_{ds} = -SP = -\frac{3}{26}$ while the initial values of Q_{ds} for nodes 3 and 5 with degree 4 are $Q_{ds} = -SP = -\frac{4}{26}$. Then, greedy Q_{ds} algorithm would first merge node 3 and node 5, which would finally lead to a single community of the whole eight nodes. However, the true community structure contains two clique communities. Accordingly, the Q_{ds} of the community structure with two clique communities, 0.4183, is larger than that of the community structure with one single large community, 0.2487. So, maximizing Q_{ds} properly should have the ability to discover the true community structure.

5.1.2 Fine-tuned Algorithm

In this part, we describe a fine-tuned community detection algorithm that iteratively improves a community quality metric M by splitting and merging the given network community structure. We denote the corresponding algorithm based

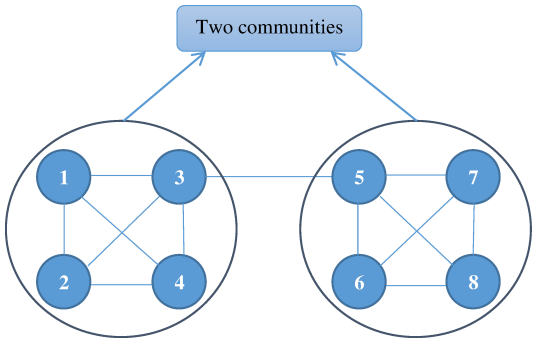


Figure 5.1: A simple network with two clique communities. Each clique has four nodes and the two clique communities are connected to each other with one single edge.

on modularity (Q) as *Fine-tuned* Q and the one based on modularity density (Q_{ds}) as *Fine-tuned* Q_{ds}. It consists of two alternating stages: the split stage and the merging stage.

In the split stage, the algorithm will split a community c into two subcommunities c_1 and c_2 based on the ratio-cut method if the split improves the value of the quality metric. The ratio-cut method [93] finds the bisection that minimizes the ratio $\frac{|E_{c_1,c_2}|}{|c_1||c_2|}$, where $|E_{c_1,c_2}|$ is the cut size (namely, the number of edges between communities c_1 and c_2), while $|c_1|$ and $|c_2|$ are sizes of the two communities. This ratio penalizes situations in which either of the two communities is small and thus favors balanced divisions over unbalanced ones. However, graph partitioning based on the ratio-cut method is a NP-complete problem. Thus, we approximate it by using the Laplacian spectral bisection method for graph partitioning introduced by Fiedler [94, 95].

First, we calculate the *Fiedler vector* which is the eigenvector of the network Laplacian matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ corresponding to the second smallest eigenvalue. Then, we put the nodes corresponding to the positive values of the *Fiedler vector* into one group and the nodes corresponding to the negative values into the other group. The subnetwork of each community is generated with the nodes and edges in that community. Although the ratio-cut approximated with spectral bisection method does allow some deviation for the sizes $|c_1|$ and $|c_2|$ to vary around the middle value, the right partitioning may not actually divide the community into two balanced or nearly balanced ones. Thus, it is to some extent inappropriate

Algorithm 1 Split_Communities(G, C)

```
1: Initialize comWeights[|C|][|C|], comEdges[|C|][|C|], and comDensities[|C|][|C|] which respectively contain #weights, #edges, and the density inside the communities and between two communities by using the network G and the community list C;
2: //Get the metric value for each community.
3: Mes[|C|] = GetMetric(C,comWeights,comDensities);
4: for i = 0 to |C| − 1 do
5:    c = C.get(i);
6:    subnet = GenerateSubNetwork(c);
7:    fiedlerVector[|c|] = LanczosMethod(subnet);
8:    nodeIds[|c|] = sort(fiedlerVector, 'descend');
9:    //Form |c| + 1 divisions and record the best one.
10:   splitTwoCom.addAll(nodeIds);
11:   for j = 0 to |c| − 1 do
12:      splitOneCom.add(nodeIds[j]);
13:      splitTwoCom.remove(nodeIds[j]);
14:      Calculate M(split) for the split at j;
15:      ΔM = M(split) − Mes[i];
16:      if ΔM(best) < ΔM (or ΔM(best) > ΔM) then
17:         ΔM(best) = ΔM;
18:         bestIdx = j;
19:      end if
20:   end for
21:   if ΔM(best) > 0 (or ΔM(best) < 0) then
22:      Clear splitOneCom and splitTwoCom;
23:      splitOneCom.addAll(nodeIds[0:bestIdx]);
24:      splitTwoCom.addAll(nodeIds[bestIdx + 1:|c| − 1);
25:      newC.add(splitOneCom);
26:      newC.add(splitTwoCom);
27:   else
28:      newC.add(c);
29:   end if
30: end for
31: return newC
```

and unrealistic for community detection problems. We overcome this problem by using the following strategies. First, we sort the elements of the *Fiedler vector* in descending order, then cut them into two communities in each of the $|c|+1$ possible ways and calculate the corresponding change of the metric values ΔM of all the $|c|+1$ divisions. Then, the one with the best value (largest or smallest depending on the measurement) of the quality metric $\Delta M(best)$ among all the $|c|+1$ divisions is recorded. We adopt this best division to the community c only when $\Delta M(best) > 0$ (or $\Delta M(best) < 0$ depending on the metric). For instance, we split the community only when $\Delta Q_{ds}(best)$ is larger than zero.

The outline of the split stage is shown in Algorithm 1. The input is a network

and a community list, and the output is a list of communities after splitting. The initialization part has $O(|E|)$ complexity. Computing *Fiedler vector* using Lanczos method [53] needs $O(|E|Kh + |V|K^2h + K^3h)$ steps, where K is the number of eigenvectors needed and h is the number of iterations required for the Lanczos method to converge. Here, K is 2 and h is typically very small although the exact number is not generally known. So, the complexity for calculating *Fiedler vector* is $O(|E| + |V|)$. Sorting the *Fiedler vector* has the cost $O(|V|log|V|)$. The search of the best division from all the $|c| + 1$ possible ones (per community c) for all the communities is achieved in $O(|E|)$ time. For the $|c| + 1$ possible divisions of a community c, each one differs from the previous one by the movement of just a single node from one group to the other. Thus, the update of the total weights, the total number of edges, and the densities inside those two split communities and between those two communities to other communities can be calculated in time proportional to the degree of that node. Thus, all nodes can be moved in time proportional to the sum of their degrees which is equal to $2|E|$. Moreover, for *Fine-tuned* Q_{ds}, computing $Q_{ds}(split)$ costs $O(|C||V|)$ because all the communities are traversed to update the *Split Penalty* for each of the $|c|+1$ divisions of each community c. All the other parts have complexity less than or at most $O(|V|)$. Thus, the computational complexity for the split stage of *Fine-tuned* Q is $O(|E| + |V|log|V|)$ while for *Fine-tuned* Q_{ds} it is $O(|E| + |V|log|V| + |C||V|)$.

In the merging stage, the algorithm will merge a community to its connected communities if the merging improves the value of the quality metric. If there are many mergers possible for a community, the one, unmerged so far, which improves the quality metric the most is chosen. Hence, each community will only be merged at most once in each stage. The outline of the merging stage is shown Algorithm 2. The input is a network and a community list, and the output is a list of communities after merging. The initialization part has the complexity $O(|E|)$. For *Fine-tuned* Q, the two "for loops" for merging any two communities have the complexity $O(|C|^2log|C|)$ because calculating $Q(merge)$ is $O(1)$ and inserting an element into the red-black tree is $O(log|C|^2) = O(2log|C|) \sim O(log|C|)$ since the maximum number of elements in the tree is $\frac{|C|(|C|-1)}{2} = O(|C|^2)$. For *Fine-tuned* Q_{ds}, the two

Algorithm 2 Merge_Communities(G, C)

```
1: Initialize comWeights[|C|][|C|], comEdges[|C|][|C|], and comDensities[|C|][|C|];
2: //Get the metric value for each community.
3: Mes[|C|] = GetMetric(C,comWeights,comDensities);
4: for i = 0 to |C| − 1 do
5:    for j = i + 1 to |C| − 1 do
6:       //Doesn't consider disconnected communities.
7:       if comWeights[i][j]==0 && comWeights[j][i]==0 then
8:          continue;
9:       end if
10:      Calculate M(merge) for merging c_i and c_j;
11:      ΔM = M(merge) − Mes[i] − Mes[j];
12:      //Record the merging information with |ΔM| descending in a red-black tree
13:      if ΔM > 0 (or ΔM < 0) then
14:         mergedInfos.put([|ΔM|, i, j]);
15:      end if
16:   end for
17: end for
18: //Merge the community with the one that improves metric's value the most
19: while mergedInfos.hasNext() do
20:    [ΔM, comId1, comId2]=mergedInfos.next();
21:    if !mergedComs.containsKey(comId1) &&
          !mergedComs.containsKey(comId2) then
22:       mergedComs.put(comId1,comId2);
23:       mergedComs.put(comId2,comId1);
24:    end if
25: end while
26: for i = 0 to |C| − 1 do
27:    c=C.get(i);
28:    if mergedComs.containsKey(i) then
29:       comId2 = mergedComs.get(i);
30:       if i < comId2 then
31:          c.addAll(C.get(comId2));
32:       end if
33:    end if
34:    newC.add(c);
35: end for
36: return newC;
```

"for loops" for merging any two communities have the complexity $O(|C|^3)$ because calculating $Q_{ds}(merge)$ needs $O(|C|)$ steps to traverse all the communities to update the *Split Penalty* and inserting an element into the red-black tree is $O(log|C|)$ as well. The other parts all have complexity at most $O(|C|^2)$. Thus, the computational complexity for the merging stage of *Fine-tuned Q* is $O(|E| + |C|^2 log|C|)$ and for the merging stage of *Fine-tuned* Q_{ds} is $O(|E| + |C|^3)$.

The fine-tuned algorithm repeatedly carries out those two alternating stages

Algorithm 3 Fine-tuned_Algorithm(G, C)

```
1: comSize = |C|;
2: splitSize = 0;
3: mergeSize = 0;
4: while comSize!=splitSize || comSize!=mergeSize do
5:     comSize = |C|;
6:     C = Split_Communities(G, C);
7:     splitSize = |C|;
8:     C=Merge_Communities(G, C);
9:     mergeSize = |C|;
10: end while
11: return C
```

until neither split nor merging can improve the value of the quality metric or until the total number of communities discovered does not change after one full iteration. Algorithm 3 shows the outline of the fine-tuned algorithm. It can detect the community structure of a network by taking a list with a single community of all the nodes in the network as the input. It can also improve the community detection results of other algorithms by taking a list with their communities as the input. Let the number of iteration of the fine-tuned algorithm be denoted as T. Then, the total complexity for *Fine-tuned Q* is $O(T(|E| + |V|log|V| + |C|^2log|C|))$ while for *Fine-tuned* Q_{ds} it is $O(T(|E| + |V|log|V| + |C||V| + |C|^3))$. Assuming that T and $|C|$ are constants, the complexity of the fine-tuned algorithms reduces to $O(|E| + |V|log|V|)$. The only part of the algorithm that would generate a non-deterministic result is the Lanczos method of calculating the *Fiedler vector*. The reason is that Lanczos method adopts a randomly generated vector as its starting vector. We solve this issue by choosing a normalized vector of the size equal to the number of nodes in the community as the starting vector for the Lanczos method. Then, community detection results will stay the same for different runs as long as the input remains the same.

5.2 Experimental Results

In this section, we first introduce several popular measures for evaluating the quality of the results of community detection algorithms. Denoting the greedy algorithm of modularity maximization proposed by Newman [36] as *Greedy Q*, we then use the mentioned above metrics to compare *Greedy Q*, *Fine-tuned Q*, and

Fine-tuned Q_{ds}. The comparison uses four real-world networks, the classical clique network and the LFR benchmark networks, each instance of which is defined with parameters each selected from a wide range of possible values. The results indicate that *Fine-tuned* Q_{ds} is the most effective method among the three, followed by *Fine-tuned* Q. Moreover, we show that *Fine-tuned* Q_{ds} can be applied to significantly improve the detection results of other algorithms.

In Subsection 2.2.2.2, we have shown that the modularity maximization approach using the eigenvectors of the Laplacian matrix is equivalent to the one using the eigenvectors of the modularity matrix. This implies that the split stage of our *Fine-tuned* Q is actually equivalent to the spectral methods. Therefore, *Fine-tuned* Q with one additional merge operation at each iteration unquestionably has better performance than the spectral algorithms. Hence, we do not discuss them here.

5.2.1 Evaluation Metrics

The quality evaluation metrics we consider here can be divided into three categories: *Variation of Information* (VI) [63] and *Normalized Mutual Information* (NMI) [96] based on information theory; *F-measure* [97] and *Normalized Van Dongen metric* (NVD) [98] based on cluster matching; *Rand Index* (RI) [99], *Adjusted Rand Index* (ARI) [100], and *Jaccard Index* (JI) [101] based on pair counting.

5.2.1.1 Information Theory Based Metrics

Given partitions C and C', *Variation of Information* (VI) [63] quantifies the "distance" between those two partitions, while *Normalized Mutual Information* (NMI) [96] measures the similarity between partitions C and C'. VI is defined as

$$\begin{aligned} VI(C, C') &= H(C) + H(C') - 2I(C, C') \\ &= H(C, C') - I(C, C'), \end{aligned} \tag{5.1}$$

where $H(.)$ is the entropy function and $I(C, C') = H(C) + H(C') - H(C, C')$ is the *Mutual Information*. Then, NMI is given by

$$NMI(C, C') = \frac{2I(C, C')}{H(C) + H(C')}. \tag{5.2}$$

Using the definitions

$$H(C) = -\sum_{c \in C} p(c) \log p(c) = -\sum_{c \in C} \frac{|c|}{|V|} \log \frac{|c|}{|V|}, \tag{5.3}$$

$$\begin{aligned} H(C, C') &= -\sum_{c \in C, c' \in C'} p(c, c') \log p(c, c') \\ &= -\sum_{c \in C, c' \in C'} \frac{|c \cap c'|}{|V|} \log \left(\frac{|c \cap c'|}{|V|} \right) \end{aligned} \tag{5.4}$$

we can express VI and NMI as a function of counts only as follows:

$$VI(C, C') = -\frac{1}{|V|} \sum_{c \in C, c' \in C'} |c \cap c'| \log \left(\frac{|c \cap c'|^2}{|c||c'|} \right), \tag{5.5}$$

$$NMI(C, C') = \frac{-2 \sum_{c \in C, c' \in C'} \frac{|c \cap c'|}{|V|} \log \left(\frac{|c \cap c'||V|}{|c||c'|} \right)}{\sum_{c \in C} \frac{|c|}{|V|} \log \frac{|c|}{|V|} + \sum_{c' \in C'} \frac{|c'|}{|V|} \log \frac{|c'|}{|V|}}, \tag{5.6}$$

where $|c|$ is the number of nodes in community c of C and $|c \cap c'|$ is the number of nodes both in community c of C and in community c' of C'.

5.2.1.2 Clustering Matching Based Metrics

Measurements based on clustering matching aim at finding the largest overlaps between pairs of communities of two partitions C and C'. *F-measure* [97] measures the similarity between two partitions, while *Normalized Van Dongen metric* (NVD) [98] quantifies the "distance" between partitions C and C'. *F-measure* is defined as

$$F\text{-}measure(C, C') = \frac{1}{|V|} \sum_{c \in C} |c| \max_{c' \in C'} \frac{2|c \cap c'|}{|c| + |c'|}. \tag{5.7}$$

NVD is given by

$$NVD(C, C') = 1 - \frac{1}{2|V|} \left(\sum_{c \in C} \max_{c' \in C'} |c \cap c'| + \sum_{c' \in C'} \max_{c \in C} |c' \cap c| \right). \tag{5.8}$$

5.2.1.3 Pair Counting Based Metrics

Metrics based on pair counting count the number of pairs of nodes that are classified (in the same community or in different communities) in two partitions

C and C'. Let a_{11} indicate the number of pairs of nodes that are in the same community in both partitions, a_{10} denote the number of pairs of nodes that are in the same community in partition C but in different communities in C', a_{01} be the number of pairs of nodes which are in different communities in C but in the same community in C', a_{00} be the number of pairs of nodes which are in different communities in both partitions. By definition, $A = a_{11} + a_{10} + a_{01} + a_{00} = \frac{|V|(|V|-1)}{2}$ is the total number of pairs of nodes in the network. Then, *Rand Index* (RI) [99] which is the ratio of the number of node pairs placed in the same way in both partitions to the total number of pairs is given by

$$RI(C, C') = \frac{a_{11} + a_{00}}{A}. \tag{5.9}$$

Denote $M = \frac{1}{A}(a_{11} + a_{10})(a_{11} + a_{01})$. Then, RI's corresponding adjusted version, *Adjusted Rand Index* (ARI) [100], is expressed as

$$ARI(C, C') = \frac{a_{11} - M}{\frac{1}{2}\left[(a_{11} + a_{10}) + (a_{11} + a_{01})\right] - M}. \tag{5.10}$$

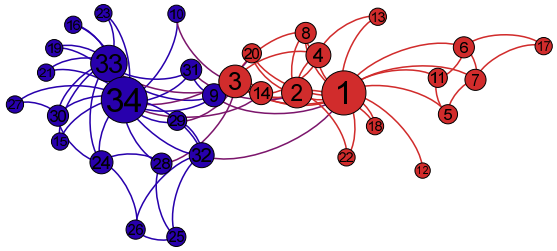
The *Jaccard Index* (JI) [101] which is the ratio of the number of node pairs placed in the same community in both partitions to the number of node pairs that are placed in the same group in at least one partition is defined as

$$JI(C, C') = \frac{a_{11}}{a_{11} + a_{10} + a_{01}}. \tag{5.11}$$

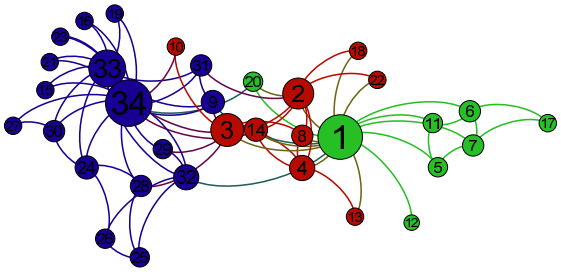
All three metrics quantify the similarity between two partitions C and C'.

5.2.2 Real-world Networks

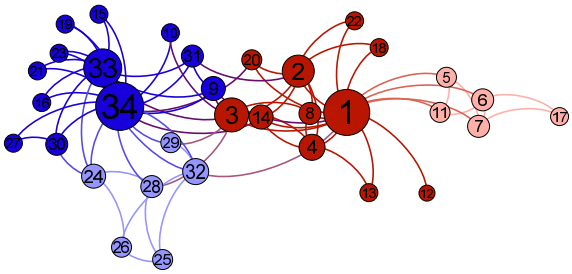
In this subsection, we first evaluate the performance of *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on two small networks (Zachary's karate club network [80] and American college football network [78]) with ground truth communities, and then on two large networks (PGP network [82] and AS level Internet) but without ground truth communities.



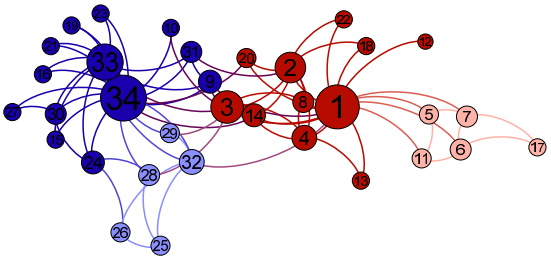
(a) Ground truth communities.



(b) Communities detected with *Greedy Q*.



(c) Communities detected with *Fine-tuned Q*.



(d) Communities detected with *Fine-tuned* Q_{ds}.

Figure 5.2: The community structure of the ground truth communities and those detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on Zachary's karate club network.

5.2.2.1 Zachary's Karate Club Network

We first compare the performance of *Greedy Q*, *Fine-tuned Q*, and *Fine-tuned* Q_{ds} on Zachary's karate club network [80]. It represents the friendships between 34 members of a karate club at a US university over a period of 2 years. During the

Table 5.1: Metric values of the community structure discovered by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on Zachary's karate club network (red italic font denotes the best value for each metric).

Algorithm	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
Greedy Q	0.3807	0.1809	***0.7677***	0.6925	***0.828***	***0.1471***	***0.8414***	***0.6803***	***0.6833***
Fine-tuned Q	***0.4198***	0.2302	0.9078	0.6873	0.807	0.1618	0.7736	0.5414	0.5348
Fine-tuned Q_{ds}	0.4174	***0.231***	0.8729	***0.6956***	0.8275	***0.1471***	0.7861	0.5669	0.5604

observation period, the club split into two clubs as a result of a conflict within the organization. The resulting two new clubs can be treated as the ground truth communities whose structure is shown in Figure 5.2(a) visualized with the opensource software Gephi [102].

Table 5.1 presents the metric values of the community structure detected by the three algorithms on this network. It shows that *Fine-tuned* Q and *Fine-tuned* Q_{ds} achieve the highest value of Q and Q_{ds}, respectively. However, most of the seven metrics based on ground truth communities imply that *Greedy* Q performs the best with only NMI and NVD indicating that *Fine-tuned* Q_{ds} has the best performance among the three algorithms. Hence, it seems that a large Q or Q_{ds} may not necessary mean a high quality of community structure, especially for Q because *Fine-tuned* Q achieves the highest Q but has the worst values of the seven metrics described in Subsection 5.2.1. We argue that the ground truth communities may not be so reasonable because *Fine-tuned* Q and *Fine-tuned* Q_{ds} in fact discover more meaningful communities than *Greedy* Q does. Figures 5.2(a)-5.2(d) show the community structure of ground truth communities and those detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds}, respectively. For results of *Greedy* Q shown in Figure 5.2(b), we could observe that there are three communities located at the left, the center, and the right side of the network. The ground truth community located on the right is subdivided into the central and right communities, but the node 10 is misclassified as belonging to the central community, while in ground truth network it belongs to community located on the left. Figure 5.2(c) demonstrates that *Fine-tuned* Q subdivides both the left and the right communities into two with six nodes separated from the left community and five nodes separated from the right community. Moreover, Figure 5.2(c) shows that *Fine-tuned* Q discovers the same

number of communities for this network as algorithms presented in [38,45,49,51]. In fact, the community structure it discovers is identical to those detected in [45,49,51]. Figure 5.2(d) shows that the community structure discovered by *Fine-tuned* Q_{ds} differs from that of *Fine-tuned* Q only on node 24 which is placed in the larger part of the left community. It is reasonable because node 24 has three connections to the larger part to which it has more attraction than to the smaller part with which it only has two connections.

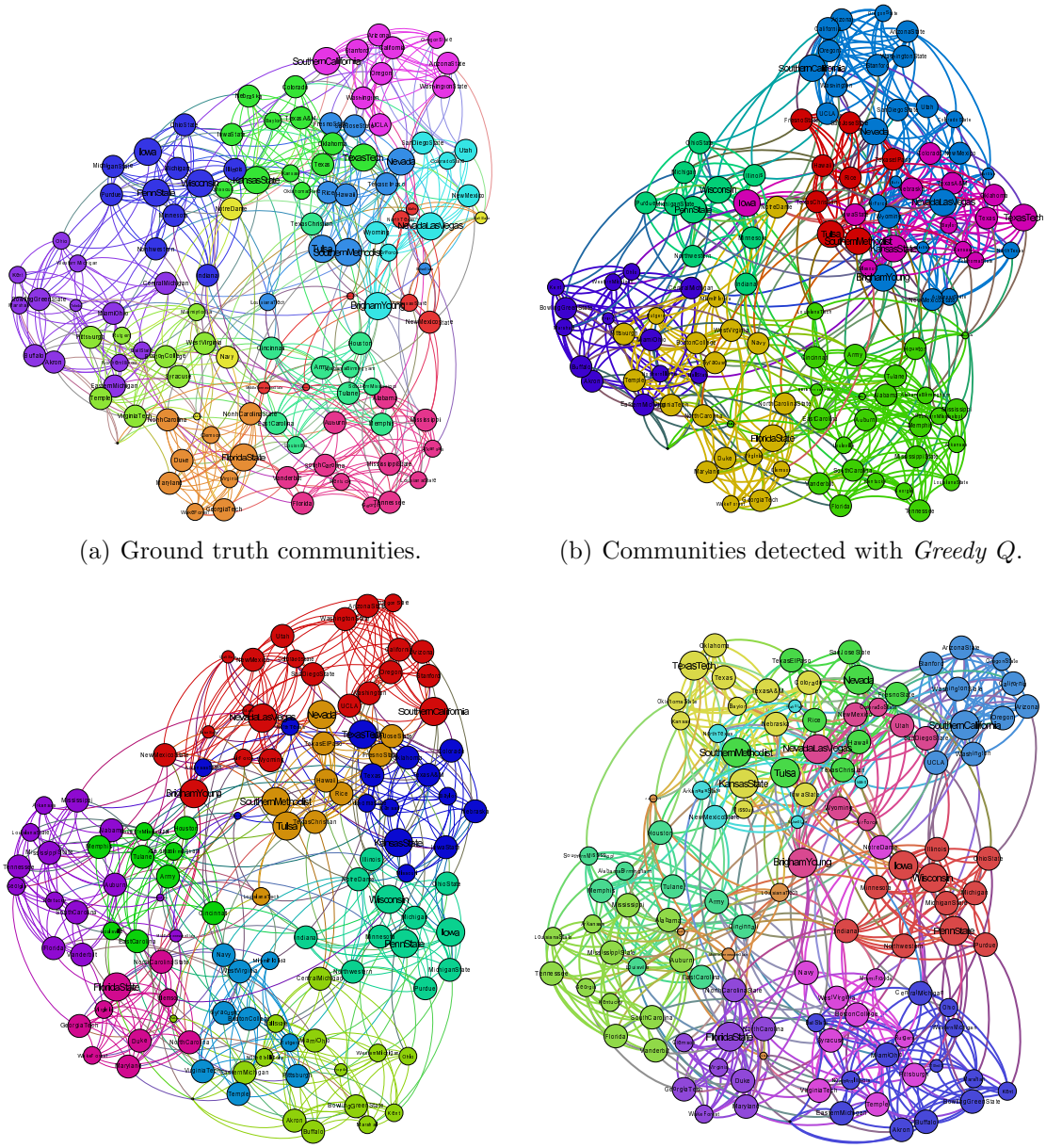
In addition, analyzing the intermediate results of *Fine-tuned* Q and *Fine-tuned* Q_{ds} reveals that the communities at the first iteration are exactly the ground truth communities, which in another way implies their superiority over *Greedy* Q. Moreover, NMI and NVD indicate that *Fine-tuned* Q_{ds} is the best among the three and all the metrics, except Q, show that *Fine-tuned* Q_{ds} performs better than *Fine-tuned* Q, supporting the claim that a higher Q_{ds} (but not Q) implies a better quality of community structure.

5.2.2.2 American College Football Network

We apply the three algorithms also to the American college football network [78] which represents the schedule of games between college football teams in a single season. The teams are divided into twelve "conferences" with intra-conference games being more frequent than inter-conference games. Those conferences could be treated as the ground truth communities whose structure is shown in Figure 5.3(a).

Table 5.2 presents the metric values of the community structure detected by the three algorithms. It shows that *Fine-tuned* Q_{ds} achieves the best values for all the nine metrics. It implies that *Fine-tuned* Q_{ds} performs best on this football network, followed by *Fine-tuned* Q. Figures 5.3(a)-5.3(d) present the community structure of ground truth communities and those discovered by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds}. Each color in the figures represents a community. It can be seen that there are twelve ground truth communities in total, seven communities detected by *Greedy* Q, nine communities discovered by *Fine-tuned* Q, and exactly twelve communities found by *Fine-tuned* Q_{ds}.

Moreover, we apply *Fine-tuned* Q_{ds} on the community detection results of



(a) Ground truth communities. (b) Communities detected with *Greedy Q*.

(c) Communities detected with *Fine-tuned Q*. (d) Communities detected with *Fine-tuned* Q_{ds}.

Figure 5.3: The community structure of the ground truth communities and those detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on American college football network.

Greedy Q and *Fine-tuned Q*. The metric values of these two community structure after improvement with *Fine-tuned* Q_{ds} are shown in Table 5.3. Compared with those of *Greedy Q* and *Fine-tuned Q* in Table 5.2, we could observe that the metric values are significantly improved with *Fine-tuned* Q_{ds}. Further, both improved community structure contain exactly twelve communities, the same number as the

Table 5.2: Metric values of the community structure detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on American college football network (red italic font denotes the best value for each metric).

Algorithm	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
Greedy Q	0.5773	0.3225	1.4797	0.7624	0.6759	0.2304	0.9005	0.5364	0.4142
Fine-tuned Q	0.5944	0.3986	0.9615	0.8553	0.8067	0.1348	0.9521	0.7279	0.6045
Fine-tuned Q_{ds}	***0.6005***	***0.4909***	***0.5367***	***0.9242***	***0.9145***	***0.07391***	***0.9847***	***0.8967***	***0.8264***

Table 5.3: Metric values of the community structure of *Greedy* Q and *Fine-tuned* Q improved with *Fine-tuned* Q_{ds} on American college football network (blue italic font indicates improved score).

Algorithm	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
Greedy Q improved with Fine-tuned Q_{ds}	***0.5839***	***0.4636***	***0.6986***	***0.9013***	***0.8961***	***0.0913***	***0.9793***	***0.8597***	***0.7714***
Fine-tuned Q improved with Fine-tuned Q_{ds}	***0.5974***	***0.4793***	***0.5096***	***0.9278***	***0.9166***	***0.06957***	***0.9837***	***0.8907***	***0.8174***

ground truth communities.

5.2.2.3 PGP Network

We then apply the three algorithms on PGP network [82]. It is the giant component of the network of users of the Pretty-Good-Privacy algorithm for secure information interchange. It has 10680 nodes and 24316 edges.

Table 5.4 presents the metric values of the community structure detected by the three algorithms. Since this network does not have ground truth communities, we only calculate Q and Q_{ds} of these discovered community structure. The table shows that *Greedy* Q and *Fine-tuned* Q_{ds} achieve the highest value of Q and Q_{ds}, respectively. It is worth to mention that the Q_{ds} of *Fine-tuned* Q_{ds} is much larger than that of *Greedy* Q and *Fine-tuned* Q, which implies that *Fine-tuned* Q_{ds} performs best on PGP network according to Q_{ds}, followed by *Greedy* Q.

5.2.2.4 AS Level Internet

The last real-world network dataset that is adopted to evaluate the three algorithms is Autonomous System (AS) level Internet. It is a symmetrized snapshot of the structure of the Internet at the level of autonomous systems, reconstructed from Border Gateway Protocol (BGP) tables posted by the University of Oregon Route Views Project. This snapshot was created by Mark Newman from data for July 22, 2006 and has not been previously published. It has 22963 nodes and 48436

Table 5.4: The values of Q and Q_{ds} of the community structure detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on PGP network (red italic font denotes the best value for each metric).

Algorithm	Q	Q_{ds}
Greedy Q	***0.8521***	0.04492
Fine-tuned Q	0.8405	0.02206
Fine-tuned Q_{ds}	0.594	***0.287***

Table 5.5: The values of Q and Q_{ds} of the community structure detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on AS level Internet (red italic font denotes the best value for each metric).

Algorithm	Q	Q_{ds}
Greedy Q	0.6379	0.002946
Fine-tuned Q	***0.6475***	0.003123
Fine-tuned Q_{ds}	0.3437	***0.03857***

edges.

Table 5.5 presents the metric values of the community structure detected by the three algorithms. Since this network does not have ground truth communities either, we only calculate Q and Q_{ds}. It can be seen from the table that *Fine-tuned* Q and *Fine-tuned* Q_{ds} achieve the highest value of Q and Q_{ds}, respectively. Moreover, the Q_{ds} of *Fine-tuned* Q_{ds} is much larger than that of *Greedy* Q and *Fine-tuned* Q, which indicates that *Fine-tuned* Q_{ds} performs best on AS level Internet according to Q_{ds}, followed by *Fine-tuned* Q.

5.2.3 Synthetic Networks

5.2.3.1 Clique Network

We now apply the three algorithms to the classical network example [11], displayed in Figure 3.3, which illustrates modularity (Q) has the resolution limit problem. It is a ring network comprised of thirty identical cliques, each of which has five nodes and they are connected by single edges. It is intuitively obvious that each clique forms a single community.

Table 5.6 presents the metric values of the community structure detected by the three algorithms. It shows that *Greedy* Q and *Fine-tuned* Q have the same

Table 5.6: Metric values of the community structure detected by *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds} on the classical clique network (red italic font denotes the best value for each metric).

Algorithm	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
Greedy Q	***0.8871***	0.46	0.9333	0.8949	0.6889	0.2333	0.9687	0.6175	0.4615
Fine-tuned Q	***0.8871***	0.46	0.9333	0.8949	0.6889	0.2333	0.9687	0.6175	0.4615
Fine-tuned Q_{ds}	0.8758	***0.8721***	***0***	***1***	***1***	***0***	***1***	***1***	***1***

Table 5.7: Metric values of the community structure of *Greedy* Q and *Fine-tuned* Q improved with *Fine-tuned* Q_{ds} on the classical clique network (blue italic font indicates improved score).

Algorithm	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
Greedy Q improved with Fine-tuned Q_{ds}	0.8758	***0.8721***	***0***	***1***	***1***	***0***	***1***	***1***	***1***
Fine-tuned Q improved with Fine-tuned Q_{ds}	0.8758	***0.8721***	***0***	***1***	***1***	***0***	***1***	***1***	***1***

performance. They both achieve the highest value of Q but get about half of the value of Q_{ds} of what *Fine-tuned* Q_{ds} achieves. In fact, *Fine-tuned* Q_{ds} finds exactly thirty communities with each clique being a single community. In contrast, *Greedy* Q and *Fine-tuned* Q discover only sixteen communities with fourteen communities having two cliques and the other two communities having a single clique. Also, we take the community detection results of *Greedy* Q and *Fine-tuned* Q as the input to *Fine-tuned* Q_{ds} to try to improve those results. The metric values of the community structure after improvement with *Fine-tuned* Q_{ds} are recorded in Table 5.7. This table shows that the community structure discovered are identical to that of *Fine-tuned* Q_{ds}, which means that the results of *Greedy* Q and *Fine-tuned* Q are dramatically improved with *Fine-tuned* Q_{ds}. Therefore, it can be concluded from Tables 5.6 and 5.7 that a larger value of Q_{ds} (but not Q) implies a higher quality of the community structure. Moreover, Q_{ds} resolves the resolution limit problem of Q. Finally, *Fine-tuned* Q_{ds} is effective in maximizing Q_{ds} and in finding meaningful community structure.

5.2.3.2 LFR Benchmark Networks

To further compare the performance of *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds}, we choose the LFR (named after the initials of names of authors) benchmark networks [103] which have become a standard in the evaluation of the per-

Table 5.8: Metric values of the community structure of *Greedy* Q on the LFR benchmark networks with $(\gamma, \beta) = (2, 1)$.

μ	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
0.05	***0.9021***	***0.4481***	***0.2403***	***0.9767***	***0.9382***	***0.0399***	***0.9959***	***0.9308***	***0.8758***
0.1	***0.8461***	0.3546	0.5213	***0.9482***	0.8539	0.0912	0.9882	0.821	0.7089
0.15	0.7862	0.2604	0.8537	0.9125	0.7604	0.1432	0.9776	0.7042	0.5573
0.2	***0.7256***	0.1934	1.3601	0.8579	0.6314	0.2173	0.9601	0.5445	0.3911
0.25	0.6612	0.1411	1.7713	0.8093	0.5477	0.2642	0.9444	0.4498	0.309
0.3	0.5959	0.09377	2.1758	0.7493	0.4745	0.3085	0.921	0.3779	0.255
0.35	0.545	0.07237	2.4599	0.7122	0.4182	0.3347	0.9045	0.3206	0.2134
0.4	0.4857	0.05521	2.7444	0.672	0.3745	0.3623	0.8874	0.2766	0.1836
0.45	0.4356	0.04133	3.0108	0.6289	0.327	0.3875	0.8617	0.2288	0.153
0.5	***0.3803***	0.03016	3.4296	0.5685	0.2874	0.4159	0.8386	0.1885	0.1282

Table 5.9: Metric values of the community structure of *Fine-tuned* Q on the LFR benchmark networks with $(\gamma, \beta) = (2, 1)$.

μ	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
0.05	0.8411	0.3875	0.8674	0.8868	0.8137	0.1049	0.9404	0.7503	0.6673
0.1	0.8419	0.3837	***0.5195***	0.9481	0.8851	0.0695	0.9875	0.8333	0.7408
0.15	***0.7886***	0.3324	***0.6453***	0.9358	0.8664	0.0844	0.9858	0.801	0.6921
0.2	0.7221	0.2922	0.9615	0.9022	0.8056	0.1222	0.9725	0.7099	0.6061
0.25	***0.6694***	0.2502	1.11	0.8833	0.7831	0.137	0.9594	0.7045	0.5939
0.3	***0.626***	0.2022	***1.0722***	***0.892***	0.813	0.1265	0.9811	0.7317	0.5963
0.35	***0.5479***	0.1516	1.6786	0.8153	0.705	0.1942	0.949	0.5963	0.4629
0.4	0.5044	0.124	1.8382	0.8108	0.6935	0.2111	0.9646	0.5592	0.4118
0.45	0.4274	0.07865	2.5657	0.7274	0.5913	0.2863	0.9463	0.4419	0.3129
0.5	0.3766	0.05808	3.0333	0.675	0.5328	0.3375	***0.9366***	0.3721	0.2537

Table 5.10: Metric values of the community structure of *Fine-tuned* Q_{ds} on the LFR benchmark networks with $(\gamma, \beta) = (2, 1)$.

μ	Q	Q_{ds}	VI	NMI	F-$measure$	NVD	RI	ARI	JI
0.05	0.845	0.4257	0.8112	0.9186	0.8564	0.09585	0.9736	0.691	0.5717
0.1	0.7934	***0.4144***	0.5809	0.9447	***0.9326***	***0.0625***	***0.9915***	***0.8566***	***0.7646***
0.15	0.7426	***0.3605***	0.6769	***0.9359***	***0.9172***	***0.0711***	***0.9902***	***0.8303***	***0.7225***
0.2	0.6786	***0.337***	***0.7824***	***0.9278***	***0.9195***	***0.0795***	***0.9908***	***0.8186***	***0.7037***
0.25	0.6202	***0.2891***	***1.0244***	***0.9046***	***0.8909***	***0.106***	***0.9868***	***0.7575***	***0.6253***
0.3	0.5693	***0.235***	1.1347	0.8919	***0.8874***	***0.1183***	***0.9845***	***0.7372***	***0.5983***
0.35	0.5443	***0.2244***	***0.9401***	***0.9123***	***0.9129***	***0.09585***	***0.989***	***0.7984***	***0.6783***
0.4	***0.505***	***0.1964***	***0.9444***	***0.9123***	***0.9091***	***0.0966***	***0.989***	***0.7929***	***0.668***
0.45	***0.4536***	***0.1632***	***1.1523***	***0.8925***	***0.8806***	***0.1196***	***0.9834***	***0.7337***	***0.6021***
0.5	0.3563	***0.1196***	***1.9677***	***0.8036***	***0.7489***	***0.2076***	0.9213	***0.4984***	***0.3813***

formance of community detection algorithms and also have known ground truth communities. The LFR benchmark network that we used here has 1000 nodes with average degree 15 and maximum degree 50. The exponent γ for the degree sequence

Table 5.11: Metric values of the community structure of *Greedy* Q improved with *Fine-tuned* Q_{ds} on the LFR benchmark networks with $(\gamma, \beta) = (2, 1)$.

μ	Q	Q_{ds}	VI	NMI	$F\text{-}measure$	NVD	RI	ARI	JI
0.05	0.8743	***0.4979***	***0.2131***	***0.98***	***0.9784***	***0.02195***	***0.997***	***0.943***	***0.895***
0.1	0.8246	***0.4522***	***0.2428***	***0.9773***	***0.9762***	***0.02395***	***0.9967***	***0.9379***	***0.8864***
0.15	0.7716	***0.4013***	***0.2972***	***0.9722***	***0.9719***	***0.0289***	***0.9962***	***0.9269***	***0.8674***
0.2	0.7232	***0.384***	***0.3503***	***0.9679***	***0.9664***	***0.03505***	***0.9959***	***0.9163***	***0.8496***
0.25	***0.6667***	***0.3347***	***0.4474***	***0.9592***	***0.9582***	***0.04485***	***0.9953***	***0.9011***	***0.8243***
0.3	***0.6094***	***0.2619***	***0.6061***	***0.9432***	***0.9457***	***0.05905***	***0.9934***	***0.876***	***0.7856***
0.35	***0.5584***	***0.2377***	***0.691***	***0.9364***	***0.94***	***0.0697***	***0.9931***	***0.8615***	***0.7626***
0.4	***0.5062***	***0.199***	***0.8285***	***0.9236***	***0.9247***	***0.0823***	***0.9916***	***0.8376***	***0.7281***
0.45	***0.4587***	***0.169***	***0.9016***	***0.9172***	***0.9222***	***0.0904***	***0.9914***	***0.8252***	***0.7099***
0.5	***0.4014***	***0.1385***	***1.2004***	***0.8906***	***0.8938***	***0.1215***	***0.9885***	***0.7686***	***0.6326***

Table 5.12: Metric values of the community structure of *Fine-tuned* Q improved with *Fine-tuned* Q_{ds} on the LFR benchmark networks with $(\gamma, \beta) = (2, 1)$.

μ	Q	Q_{ds}	VI	NMI	$F\text{-}measure$	NVD	RI	ARI	JI
0.05	***0.8519***	***0.4463***	***0.5949***	***0.937***	***0.8954***	***0.0709***	***0.9781***	***0.8177***	***0.7377***
0.1	0.8186	***0.4397***	***0.3405***	***0.9679***	***0.9615***	***0.03415***	***0.9952***	***0.9125***	***0.8452***
0.15	0.769	***0.391***	***0.4285***	***0.9597***	***0.9533***	***0.0432***	***0.9946***	***0.8993***	***0.8231***
0.2	0.7185	***0.369***	***0.4654***	***0.9571***	***0.9479***	***0.04975***	***0.9943***	***0.8853***	***0.8014***
0.25	0.6672	***0.326***	***0.5667***	***0.9477***	***0.9365***	***0.05805***	***0.9936***	***0.8713***	***0.7785***
0.3	0.6109	***0.2598***	***0.6962***	***0.9346***	***0.9372***	***0.06505***	***0.9926***	***0.8609***	***0.762***
0.35	0.5474	***0.2297***	***0.9525***	***0.9108***	***0.9175***	***0.0961***	***0.9882***	***0.7963***	***0.6821***
0.4	0.4966	***0.1983***	***1.0601***	***0.9021***	***0.9118***	***0.1029***	***0.9896***	***0.7963***	***0.672***
0.45	***0.4284***	***0.1535***	***1.4754***	***0.8635***	***0.8694***	***0.1486***	***0.9831***	***0.6836***	***0.5362***
0.5	0.3654	***0.1258***	***1.9271***	***0.8192***	***0.8193***	***0.1987***	***0.968***	***0.5852***	***0.4423***

varies from 2 to 3. The exponent β for the community size distribution ranges from 1 to 2. Then, four pairs of the exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1),$ and $(3, 2)$ are chosen in order to explore the widest spectrum of graph structures. The mixing parameter μ is varied from 0.05 to 0.5. It means that each node shares a fraction $(1 - \mu)$ of its edges with the other nodes in its community and shares a fraction μ of its edges with the nodes outside its community. Thus, low mixing parameters indicate strong community structure. Also, we generate 10 network instances for each μ. Hence, each metric value in Tables 5.8-5.12 represents the average metric values of all 10 instances. Since the experimental results are similar for all four pairs of exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1),$ and $(3, 2)$, for the sake of brevity, we only present the results for $(\gamma, \beta) = (2, 1)$ here.

Tables 5.8-5.10 show the metric values of the community structure detected with *Greedy* Q, *Fine-tuned* Q, and *Fine-tuned* Q_{ds}, respectively, on the LFR benchmark networks with $(\gamma, \beta) = (2, 1)$ and μ varying from 0.05 to 0.5. The red italic font in the table denotes that the corresponding algorithm achieves the best value for a certain quality metric among the three algorithms. The results in these tables show that *Greedy* Q obtains the best values for all the nine measurements when $\mu = 0.05$, while *Fine-tuned* Q_{ds} achieves the highest values of Q_{ds} and the best values for almost all the seven metrics based on ground truth communities when μ ranges from 0.1 to 0.5. Also, *Fine-tuned* Q gets the second best values for Q_{ds} and almost all the seven metrics in the same range of μ. However, for Q the best is *Greedy* Q, followed by *Fine-tuned* Q, and *Fine-tuned* Q_{ds} is the last.

In summary, the seven measurements based on ground truth communities are all consistent with Q_{ds}, but not consistent with Q. This consistency indicates the superiority of Q_{ds} over Q as a community quality metric. In addition, *Fine-tuned* Q_{ds} performs best among the three algorithms for $\mu > 0.05$, which demonstrates that it is very effective and does a very good job in optimizing Q_{ds}.

We then take the community detection results of *Greedy* Q and *Fine-tuned* Q as the input to *Fine-tuned* Q_{ds} to improve those results. The measurement values of the community structure after improvement with *Fine-tuned* Q_{ds} are displayed in Tables 5.11 and 5.12. The blue italic font in Table 5.11 and Table 5.12 implies that the metric value in these two tables is improved compared to the one in Table 5.8 and that in Table 5.9, respectively. Then, compared with those of *Greedy* Q shown in Table 5.8 and those of *Fine-tuned* Q shown in Table 5.9, all measurements, except in some cases for Q, are significantly improved with *Fine-tuned* Q_{ds}. This again indicates that all the seven metrics described in Subsection 5.2.1 are consistent with Q_{ds}, but not consistent with Q. Interestingly, those results are even better than those of *Fine-tuned* Q_{ds} itself presented in Table 5.10. Thus, it can be concluded that *Fine-tuned* Q_{ds} is very powerful in improving the community detection results of other algorithms.

5.3 Summary

In this chapter, we propose an efficient and effective fine-tuned algorithm to maximize Q_{ds}. This new algorithm can actually be used to optimize any community quality metric. We evaluate the three algorithms, *Greedy* Q, *Fine-tuned* Q based on Q, and *Fine-tuned* Q_{ds} based on Q_{ds}, with seven metrics based on ground truth communities. These evaluations are done on four real-world networks, and also on the classical clique network and the LFR benchmark networks, each instance of the last is defined with parameters selected from wide range of their values. The results demonstrate that *Fine-tuned* Q_{ds} performs best among the three algorithms, followed by *Fine-tuned* Q. The experiments also show that *Fine-tuned* Q_{ds} can dramatically improve the community detection results of other algorithms. In addition, all the seven quality metrics based on ground truth communities are consistent with Q_{ds}, but not consistent with Q, which indicates the superiority of Q_{ds} over Q as a community quality metric.

CHAPTER 6
APPLICATION OF NEW METRICS FOR COMMUNITY STRUCTURE WITH LINK PREDICTION RANKING

Many networks, including Internet, citation networks, transportation networks, e-mail networks, and social and biochemical networks, display community structure which identifies groups of nodes with more connections inside the group than outside [5]. Detecting and characterizing such a community structure, which is known as community detection, is one of the fundamental techniques of network science. Community detection has been shown to reveal latent yet meaningful structure in networks such as groups in online and contact-based social networks, functional modules in protein-protein interaction networks, groups of customers with similar interests in online retailer user networks, groups of scientists in interdisciplinary collaboration networks, etc. [6].

Therefore, it is expected that many networks, like social and biology networks, have highly modular subsets or, in other words, community structure. However, the community structure discovered by community detection algorithms does not usually represent the reality. The primary reason for this is that available network datasets are often incomplete and inaccurate. For example, in the process of collecting, gathering, or recording information from online social networks, some data can be lost or incorrect because of complex relations between individuals, privacy constraints, improper or imprecise sampling methods, etc. Also, in a network representing interactions between genes in some species edges are typically determined experimentally, so the number of known edges may be much smaller than in reality. Moreover, random spatial collocation of some genes may be wrongly interpreted as an active interaction. Consequently, the networks we derive from available data usually have some noise, like missing some edges or having some incorrectly identified so extraneous edges, which may cause the collected datasets to appear less modular

Portions of this chapter have been submitted as: M. Chen, A. Bahulkar, K. Kuzmin, and B. K. Szymanski, "Improving network community structure with link prediction ranking," in *Proc. 7th Workshop Complex Networks* (under review), 2016.

than the underlying networks really are.

Thus, the purpose of this chapter is to propose and evaluate methods of recovering or improving the network community structure which may be hidden or impaired by missing or extraneous edges. Our goal is to make the network more modular by recovering missing edges and removing extraneous edges. We introduce a link improvement procedure that removes a certain fraction of existing low ranking links and replaces them with potential links (e.g., the links of the complete graph with the same set of nodes as the current graph that do not exist in the current graph) ranked highly by a link prediction metric, such as the number of common neighbors (CN), Adamic-Adar (AA) [104], or PropFlow (PF) [105]. We evaluate this link improvement approach on seven real-world network datasets, including two friendship networks, two collaboration networks, and a co-purchasing network. Table 6.1 shows the basic properties of these datasets. The networks vary in size from dozens of nodes to hundreds of thousands of nodes. After enhancing the networks with our link improvement procedure, we run community detection algorithm SpeakEasy [74] to detect communities on them and then measure the quality of the discovered community structure with eight community quality metrics, including two global metrics, modularity [5] and modularity density [21, 22, 91, 106, 107], and six local metrics.

The results show that the community structure of five out of seven real-world networks is significantly refined with our link improvement method. We observed that the best improvement of network community structure was delivered by common neighbors metric, followed by Adamic-Adar. PropFlow works extremely well on and only on Gowalla dataset [108]. We believe the reason is that Gowalla is the only network which is sensitive to geo-distances between nodes, and PropFlow is a geodesic proximity measure [105]. This demonstrates that a single link prediction metric cannot perform equally well on all networks. It follows that the performance of link prediction and improvement methods depends on the meaning of the relationships which define links in the network. Moreover, all three link prediction metrics perform poorly on Karate [80] and Santafe [78] datasets. The reason is that these two networks are very small and their members have been together in the same club

or institute for such a long time that they were able to evolve their links to the desired state. However, the other five networks are different. They are either very large or their members do not have a global knowledge about the network structure or all other nodes. Consequently, each of these networks has many potentially good matches that are not realized even after evolving over a long period of time [109].

6.1 Related Work

An observation that many real-world datasets are not a complete and accurate representation of their respective underlying networks was made by Yan and Gregory in [110]. They proposed a classification of reasons for missing edges which is helpful in selecting the most suitable link predictor. However, the opposing case when a dataset might have extra edges (e.g., caused by noisy instrumentation or collection methods) which are not part of the underlying network was not considered.

The performance of different link prediction methods is assessed in [110] using a partial network which is simulated by deleting a certain fraction of edges from the input dataset. Then one of the predictors is applied to the network. A set of considered predictors comprises CN, Jaccard, meet/min, geometric, AA, resource allocation index (RA), preferential attachment (PA), hierarchical structure method (HRG), and stochastic block model (BM). Finally, the results are compared with the original network based on the value of the area under the ROC (receiver-operating characteristic) curve (AUC). The quality of the resulting communities is compared to their true versions which are assumed to be available. Normalized mutual information (NMI) [91] is used as a measure of community quality.

In [111] Yan and Gregory designed a framework which guides link prediction methods based on the results of community detection. The key assumption behind this approach is that nodes within the same community are more similar than nodes located in different communities. Therefore, intra-community edges suggested by a link predictor are added to the network first, followed by inter-community edges. Experimental verification was performed on the LFR benchmark [103] and six very small real-world networks using several link predictors (CN, Jaccard, AA, RA, PA, and HRG) and two community detection algorithms.

A more detailed analysis of the relation between community structure and link formation is given in [112]. A Fast Block probabilistic Model (FBM) uses a greedy strategy to sample the space of possible network partitions. Given an array of communities produced by FBM, the density of links inside a community and between any two communities determines the probability of adding a particular link. A further development of FBM is a local degree blocking model [113] which uses local structural information of the network for improved performance.

It is reasonable to expect that if knowing the community structure of a network is helpful in better predicting the missing links, and inverse might also be true. In other words, modifying a network according to recommendations from a link predictor might lead to a more distinct network structure and thus improve the quality of communities. Indeed, a common approach to enhancing the quality of community detection methods using link prediction techniques is to introduce a preprocessing step to ameliorate the network by reinforcing its community structure. Many researchers have come up with solutions which follow this general two-phase workflow of link prediction followed by community detection. Algorithms mainly vary in which link prediction and community detection methods are used and how their results are combined.

An example of such solution is a method proposed by Yan and Gregory in [114]. They observed that community detection and link prediction have a lot in common, as both problems use node similarity features to unveil the relations between nodes in a network. During the first phase of the algorithm, link prediction is applied to assign weights to the existing edges of a network. CN is used as a node similarity measure. This predictor is based on the local structure of the network and therefore can be efficiently computed. Only edge weights are modified but no new edge is added to the network. Then a community detection algorithm is applied to the weighted network.

The method described in [114] is rather general and can be used with any link prediction and community detection technique, and applied to unweighted, undirected, and unipartite networks. However, using appropriate link prediction and community detection methods can make this approach suitable for weighted

or bipartite cases. Experimental data are presented for the LFR benchmark and a selection of small real-world networks using five different community detection methods. The community quality is judged upon using NMI for synthetic networks and modularity for real-world ones.

A recent paper [115] of Burgess et al. proposes a more complicated solution called EdgeBoost which involves running link prediction multiple times on the same input network thus creating a family of enhanced networks. Community detection is then performed for each network from this family. The final result is constructed by aggregating community detection results of each individual network.

At the link prediction phase EdgeBoost builds a probability distribution over predicted edges. By sampling this distribution rather than simply running a link predictor, the algorithm tries to add mostly intra-community edges and thus enhance the community structure of the network. The community detection stage offers six different algorithms to choose from. Combining community detection results of individual enhanced networks into a single output partition is performed using a co-community network. This network has the same nodes as the original graph. The edges are weighted according to the normalized frequency of the number of times two nodes were assigned to the same community.

EdgeBoost is designed for small and medium scale networks. Its performance has been experimentally verified using the LFR benchmark and a selection of small real-world datasets. The implementation used in the study is limited to performing only disjoint community detection. A selection of link predictors which includes AA, CN, and Jaccard has been considered. Interestingly enough, although no significant difference was found between these three methods, Jaccard slightly outperformed its competitors and was therefore selected as the link predictor for EdgeBoost.

6.2 Link Replacing Methodology

Algorithm 4 defines our approach to ranking the existing and potential links and selecting them for removal and addition. At the first stage, every possible edge (whether existing or potential) in the network is assigned a rank based on the score

Algorithm 4 : Link ranking and replacement

Input: Graph $G = (V, E)$, link predictor $\mathcal{L}$, fraction of edges to be replaced f
Output: Graph $G' = (V, E')$ with improved community structure

```
E' ← E
Ē ← {{u, v} : ∀u ∈ V, ∀v ∈ V, s.t. u ≠ v} \ E
R_E ← ()
R_Ē ← ()
for all e ∈ E do
    Add (e, L(e)) to R_E
end for
Sort R_E in the order of ascending rank values
for all e ∈ Ē do
    Add (e, L(e)) to R_Ē
end for
Sort R_Ē in the order of descending rank values
n ← ⌊f · |E|⌋
for i = 1 to n do
    e ← edge from the i^th top tuple of R_Ē
    E' ← E' ∪ {e}
end for
for i = 1 to n do
    e ← edge from the i^th top tuple of R_E
    if ∃{u, v} ∈ E, s.t. e = {u, v} and (deg(u) = 1 or deg(v) = 1) then
        n ← n + 1
    else
        E' ← E' \ {e}
    end if
end for
```

returned by a link predictor $\mathcal{L}$. We used LPmade library [116] which is a scalable, high-performance, and cross-platform software for unsupervised link prediction and analysis. LPmade offers a comprehensive set of link predictors but we focused on three of them: CN, AA, and PF. More details about our choice of link predictors and their features are provided in Section 6.3.1. Edges and their corresponding rank scores are kept separately for existing and potential edges as required by subsequent processing logic.

During the second phase of the algorithm edge replacements take place. First, a number (defined by a certain fraction $f(\%)$ of all edges) of the top ranked potential edges are added to the network. The impact of selecting the value of parameter

f on performance is discussed in Section 6.3.5. Then, the same number of the lowest ranked existing edges are removed from the network. In order to prevent the formation of isolated nodes (i.e. nodes with a degree of 0), no edge is removed if at least one of its endpoints has a degree of 1. Instead, an existing edge with the next larger rank is selected for removal.

Although our approach follows a known two-phase pattern of enhancing the network with link prediction and then performing community detection, there are substantial differences from other solutions described in Section 6.1. In contrast to [110], our focus is not to compare the performance of different community detection methods relative to the true network which is often unknown. Instead, we concentrate on a single community detection method and consider its performance on a network enhanced with different link prediction procedures. Additionally, we assume that no sole quality metric could be used on all occasions. We experiment with different metrics (see Section 6.3.3 for details) and show (see Section 6.3.5) that depending on the dataset some metrics work better than the others. Thus, it is advisable to always apply several metrics rather than any particular one. Moreover, our method accounts for extraneous existing edges that should be removed from the dataset.

6.3 Evaluation and Analysis

In this section, we evaluate our link improvement procedure introduced in Section 6.2 on seven real-world network datasets. The evaluation is consisted of the following steps:

- First, we adopt our link improvement procedure with link prediction metrics being the number of common neighbors, Adamic-Adar [104], and PropFlow [105] on the network datasets we considered to remove a certain fraction $f(\%)$ of existing low ranking links and replace them with potential highly ranked predicted links. In our experiments, we choose $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$ where $f = 0$ means that there is no change to the original networks.
- Then, we use the community detection algorithm SpeakEasy [74] to detect the community structure of the networks generated with the link replacing procedure.

- Finally, we calculate the community quality metric values of the community structure discovered with SpeakEasy to check the performance of our link improvement method on improving the quality of network community structure.

6.3.1 Link Prediction Metrics

In this part, we introduce the link prediction metrics adopted in the experiments. Instead of exhaustively testing all the link prediction metrics proposed in the literatures, we select three local and thus computationally efficient metrics that are among the best [105, 117, 118]: common neighbors, Adamic-Adar [104], and PropFlow [105].

6.3.1.1 Common Neighbors

Common neighbors (CN) measures the number of neighbors that nodes x and y have in common. In social setting, individuals are more likely to be acquainted if they have many common acquaintances. It is defined as:

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)|, \tag{6.1}$$

where $\Gamma(x) = \{y | y \in V, (x, y) \in E\}$ is the set of neighbors of node x. The computational complexity to calculate this score for a network is $O(|E|)$.

6.3.1.2 Adamic-Adar

Adamic-Adar (AA) [104] is a refinement of common neighbors by assigning the less connected neighbors more weights instead of simply counting all equally. In social setting, it reflects the observation that being one of many acquaintances of a person, decreases the chance of being introduced to others by such a person. It is given by the following formula:

$$AA(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}, \tag{6.2}$$

The computational complexity to calculate this score for a network is $O(|E|)$.

6.3.1.3 PropFlow

PropFlow (PF) [105] measures the geodesic proximity between nodes. It represents the probability that a restricted random walk starting at x ends at y in l steps or fewer using link weights as transition probabilities. The restrictions are that the walk terminates upon reaching y or upon revisiting any node including x. This produces a score s_{xy} that can serve as an estimation of the likelihood of new links. In our experiments, we choose l to be 4 to consider neighbors of a node up to the fourth degree which also enables us to more efficiently calculate the score. The complexity to calculate this score for a network is $O(d^4|V|)$ or $O(d^3|E|)$ where d is the average node degree of the network. For a sparse network, the complexity is linear in the number of edges.

6.3.2 SpeakEasy

SpeakEasy [74] is a label propagation community detection algorithm which identifies communities using top-down and bottom-up approaches simultaneously. Specifically, nodes join communities based on their local connections and also the global information about the network structure. It adopts consensus clustering to get robust community structure. That is, the partition with the highest average adjusted Rand Index [91] among all other partitions obtained from replicate runs is selected as the representative partition. As shown in [74], SpeakEasy achieved top performance on both social and biological networks. In our experiments, we choose the number of iterations of the label propagation procedure to be 50 and conduct 20 replicate runs for consensus clustering to get more robust and deterministic results. SpeakEasy gets overlapping community structure with threshold parameter from 0 to 1 exclusive, and gets disjoint community structure with threshold to be 1. In the experiments, we set the threshold to be 1 as we only consider disjoint community structure in this chapter.

6.3.3 Community Quality Metrics

In our experiments, we adopt two global community quality metrics, modularity [5] and modularity density [21, 22, 91, 106, 107], and six local community quality

metrics to measure the quality of community structure that SpeakEasy discovers on the networks generated with the link improvement procedure described in Algorithm 4.

6.3.3.1 Newman's Modularity

Modularity [5] measures the difference between the actual fraction of edges within the community and such fraction expected in a randomized graph with the same number of nodes and the same degree sequence. For a given community partition of a unweighted and undirected network $G = (V, E)$ with $|E|$ edges, modularity (Q) is given by

$$Q = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 \right], \tag{6.3}$$

where C is the set of all the communities, c is a specific community in C, $|E_c^{in}|$ is the number of edges between nodes within community c, and $|E_c^{out}|$ is the number of edges from the nodes in community c to the nodes outside c.

6.3.3.2 Modularity Density

Modularity density [21, 22, 91, 106, 107] is proposed to simultaneously resolve the two opposite yet coexisting problems of Newman's modularity which in some cases tends to favor small communities over large ones while in others, large communities over small ones. The latter tendency is known in the literature as the resolution limit problem of modularity [11]. Modularity density introduces two additional components, split penalty and community density, into Newman's modularity given in Equation (6.3). Split penalty is the fraction of edges that connect nodes of different communities. Community density includes internal community density and pair-wise community density. The definition of modularity density (Q_{ds}) for

unweighted and undirected networks is given by:

$$Q_{ds} = \sum_{c \in C} \left[\frac{|E_c^{in}|}{|E|} d_c - \left(\frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} d_c \right)^2 - \sum_{\substack{c' \in C \\ c' \neq c}} \frac{|E_{c,c'}|}{2|E|} d_{c,c'} \right], \tag{6.4}$$

$$d_c = \frac{2|E_c^{in}|}{|c|(|c|-1)}, d_{c,c'} = \frac{|E_{c,c'}|}{|c||c'|},$$

where d_c is the internal density of community c, $|E_{c,c'}|$ is the number of edges from c to c', and $d_{c,c'}$ is the pair-wise density between community c and community c'.

6.3.3.3 Six Local Community Quality Metrics

Modularity and modularity density are two global community quality metrics. We also consider six local community quality metrics: *Intra-density*, *Contraction*, *Expansion*, *Conductance* [21, 22, 69, 119], the *Fitness* function [70], and the *Average Modularity Degree* [71]. These metrics describe how the connectivity structure of a given set of nodes resembles a community. All of them rely on the intuition that communities are sets of nodes with many edges inside them and few edges outside of them.

***Intra-density* (*ID*):** d_c in Equation (6.4). The larger the value of this metric, the higher the community quality.

***Contraction* (*CNT*):** $2|E_c^{in}|/|c|$; it measures the average number of edges per node inside the community c. The larger value of contraction means the higher community quality.

***Expansion* (*EXP*):** $|E_c^{out}|/|c|$; it measures the average number of edges (per node) that point outside the community c. A smaller value of expansion corresponds to a better community structure.

***Conductance* (*CND*):** $\frac{|E_c^{out}|}{2|E_c^{in}|+|E_c^{out}|}$; it measures the fraction of the total number of edges that point outside the community. A smaller value of conductance means higher community quality.

The *Fitness* (*F*) function: $\frac{|E_c^{in}|}{|E_c^{in}|+|E_c^{out}|}$; it is the ratio between the internal degree and the total degree of a community c. A larger value of F indicates better com-

Table 6.1: Basic properties of the real-world network datasets used in the experiments.

Name	#Nodes	#Edges	Description
Gowalla	391,222	2,176,188	Gowalla friendship net. [108]
Amazon	334,863	925,872	Amazon product net. [33]
DBLP	317,080	1,049,866	DBLP collaboration net. [33]
Santafe	118	200	Santa Fe collaboration net. [78]
Football	115	613	College football net. [78]
Dolphin	62	159	Dolphin social net. [76]
Karate	34	78	Zachary's karate club net. [80]

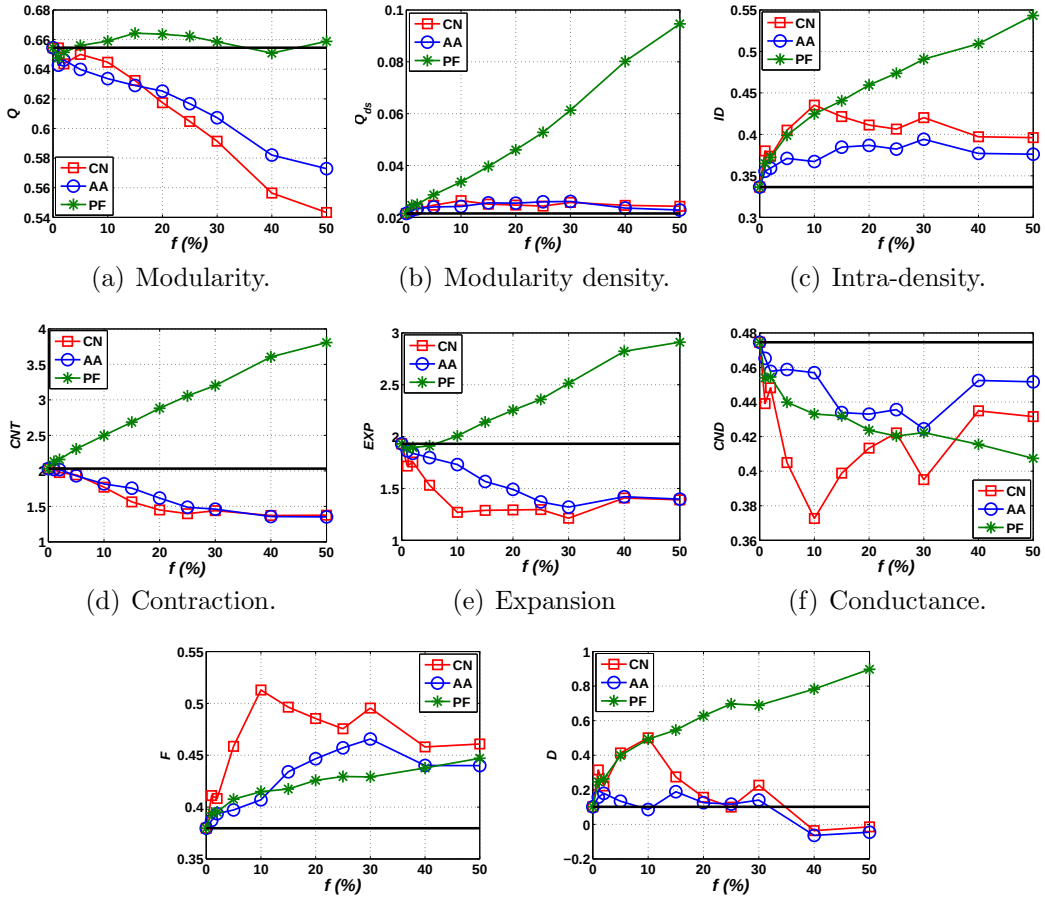
munity structure.

Average Modularity Degree (D): $\sum_{c \in C} \frac{2|E_c^{in}| - |E_c^{out}|}{|c|}$; it is the summation of the average modularity degree of each community. The average modularity degree of a community ($\frac{2|E_c^{in}| - |E_c^{out}|}{|c|}$) equals to the average inner degree ($\frac{2|E_c^{in}|}{|c|}$) minus the average outer degree ($\frac{|E_c^{out}|}{|c|}$). The larger the value of D, the higher the community quality.

6.3.4 Dataset Description

We consider totally seven real-world network datasets, including two friendship networks, two collaboration networks, and a co-purchasing network. Table 6.1 shows the basic properties of all these datasets. The networks vary in size from dozens of nodes to hundreds of thousands of nodes.

Gowalla, collected from a location-based social networking provider called Gowalla, includes 391,222 users with public profiles (friends and checkins) from mid September in 2011 to late October of that year [108]. Edges in this network indicate friendships between users. **Amazon** is a product co-purchasing network of the Amazon website [33]. The nodes of the network represent products and edges link commonly co-purchased products. **DBLP** is a scientific collaboration network where nodes represent authors and edges connect authors that have co-authored a paper [33]. **Santafe** is the largest connected component of the collaboration network of scientists at the Santa Fe Institute during years 1999 and 2000 [78]. **Football** is a network that represents the schedule of games between college football teams in a single season [78]. **Dolphin** is a social network of frequent associations between



(a) Modularity. (b) Modularity density. (c) Intra-density.

(d) Contraction. (e) Expansion (f) Conductance.

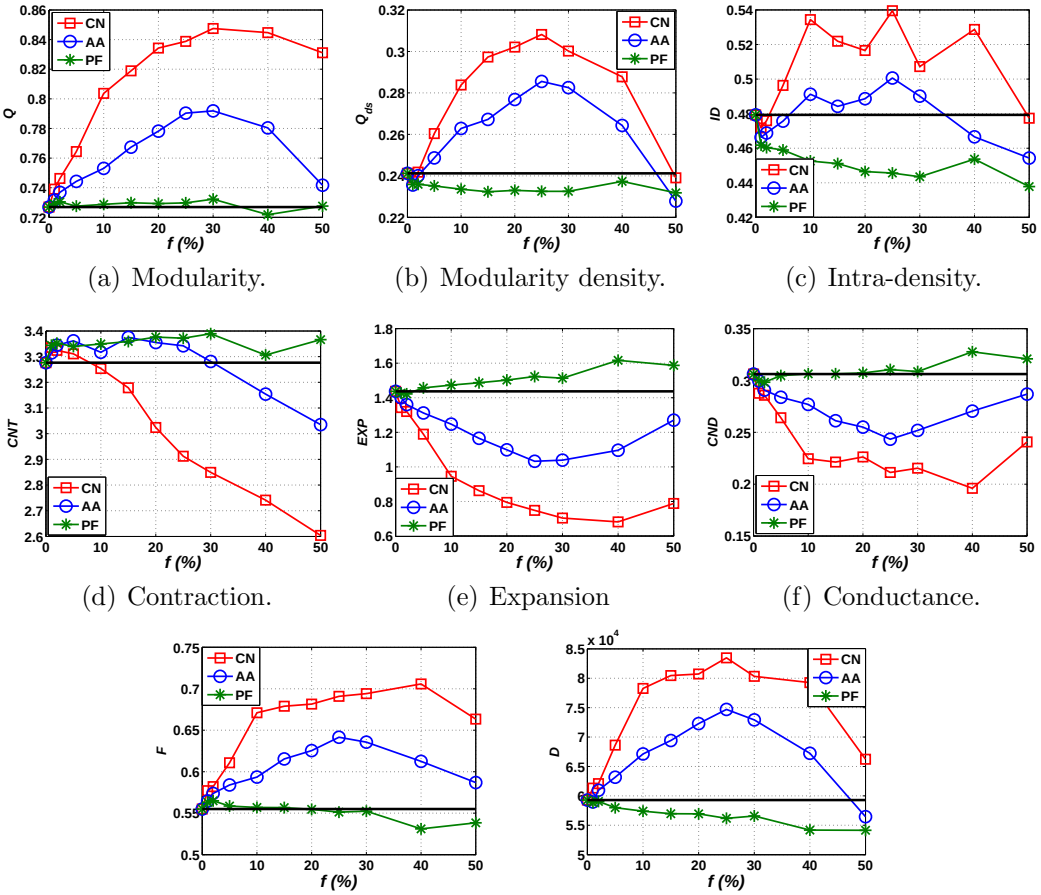
(g) Fitness. (h) Average modularity degree.

Figure 6.1: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from Gowalla using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$**.**

62 dolphins in a community living off Doubtful Sound, New Zealand [76]. **Karate** is a network representing the friendships between 34 members of a karate club at a US university during two years [80].

6.3.5 Experimental Results

In this part, we show the calculated community quality metric values of the community structure that SpeakEasy detects on the networks generated from the seven real-world network datasets by using our link improvement procedure with the replacing fraction $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$ where $f = 0$ means that there is no change to the original networks.



(a) Modularity. (b) Modularity density. (c) Intra-density.

(d) Contraction. (e) Expansion (f) Conductance.

(g) Fitness. (h) Average modularity degree.

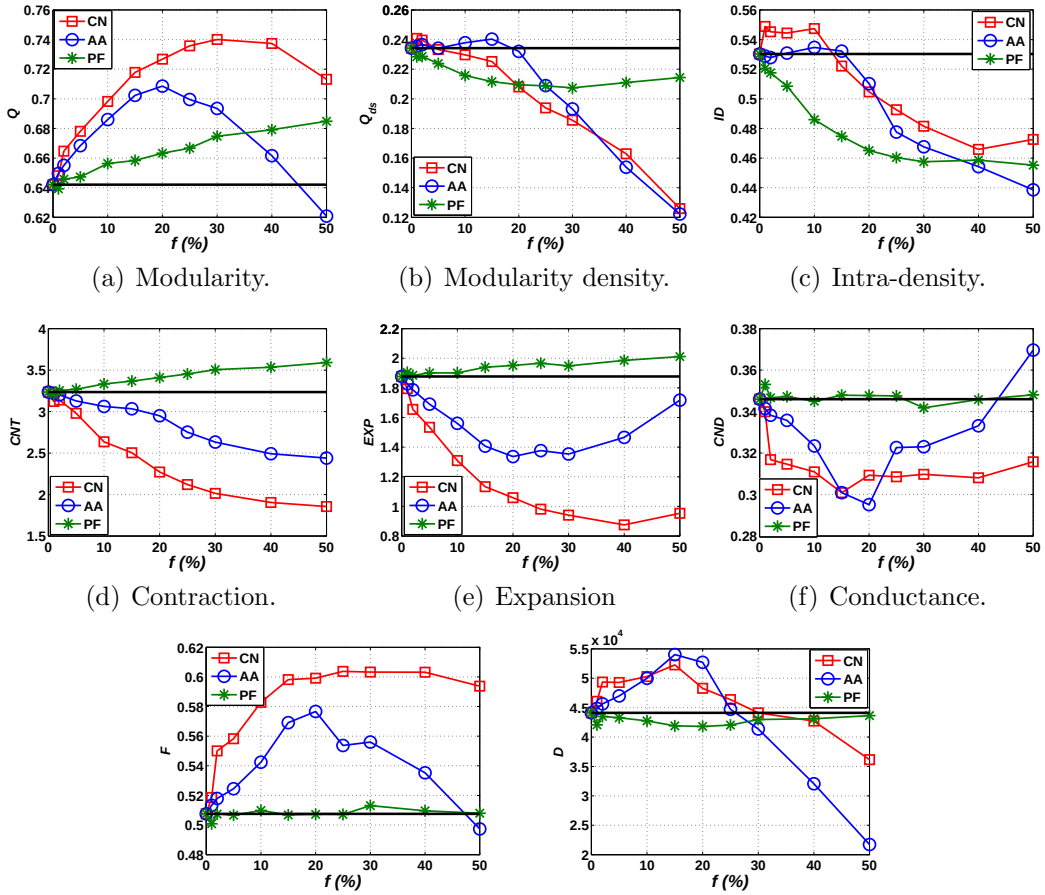
Figure 6.2: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from Amazon using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$**.**

Figure 6.1 shows the results for Gowalla dataset. The horizontal line indicates the metric values of the community structure detected on the original unchanged network. It can be seen that our link improvement method with common neighbors, Adamic-Adar, and PropFlow improve the quality of the community structure significantly according to the eight community quality metrics. With common neighbors, the maxima or minima (for expansion and conductance) of the community quality metrics are achieved at $f = 10$, except for modularity and contraction. With Adamic-Adar, the maxima or minima are obtained at $f = 30$, with exception being again modularity and contraction. PropFlow performs extremely well on Gowalla, except with expansion. The improvement goes beyond $f = 50$, excluding modulari-

ty which reaches its maximum at $f = 15$. Thus, we can observe that there is a limit or threshold (varying for different link prediction metrics) of how many links could be replaced for the purpose of improving community structure of a network. Going beyond this threshold may lead to higher cost and lower performance. However, one point worth mentioning is that even going beyond the threshold, the quality of the obtained community structure may still be better than the quality of the original community structure yet lower than the optimal one. We could also notice that common neighbors metric generally performs better than Adamic-Adar and in some cases better than PropFlow on Gowalla dataset. In most cases, PropFlow has the best performance, especially according to modularity, modularity density, intra-density, contraction, and average modularity degree. Figure 6.1(b) shows that the value of modularity density grows up by about two magnitudes of the original value with PropFlow. However, as shown later, PropFlow does not work well on the other six networks. We believe the reason is that Gowalla is the only network among the seven that is sensitive to geo-distances between nodes, and PropFlow is a geodesic proximity measure [105].

Figure 6.2 presents the results of Amazon dataset. The horizontal line again shows the metric values of the community structure detected on the original unchanged network. We can observe that PropFlow performs poorly on this dataset, except that modularity and contraction increase slightly from $f = 0$ to $f = 30$. In contrast, common neighbors and Adamic-Adar work well on this network. With Adamic-Adar, almost all the quality metrics reach their maxima or minima at $f = 25$; with common neighbors, modularity gets its maximum at $f = 30$, modularity density, intra-density, and average modularity degree attain their maxima at $f = 25$, while expansion, conductance, and fitness function achieve their minima or maximum at $f = 40$. Thus, there is a threshold of how many links could be replaced on Amazon too. We can also notice that common neighbors metric performs much better than Adamic-Adar and PropFlow.

Figure 6.3 displays the results of DBLP. It shows that with PropFlow our link improvement approach improves the quality of community structure according to modularity (Figure 6.3(a)) and contraction (Figure 6.3(d)), while actually impairs



(a) Modularity. (b) Modularity density. (c) Intra-density.

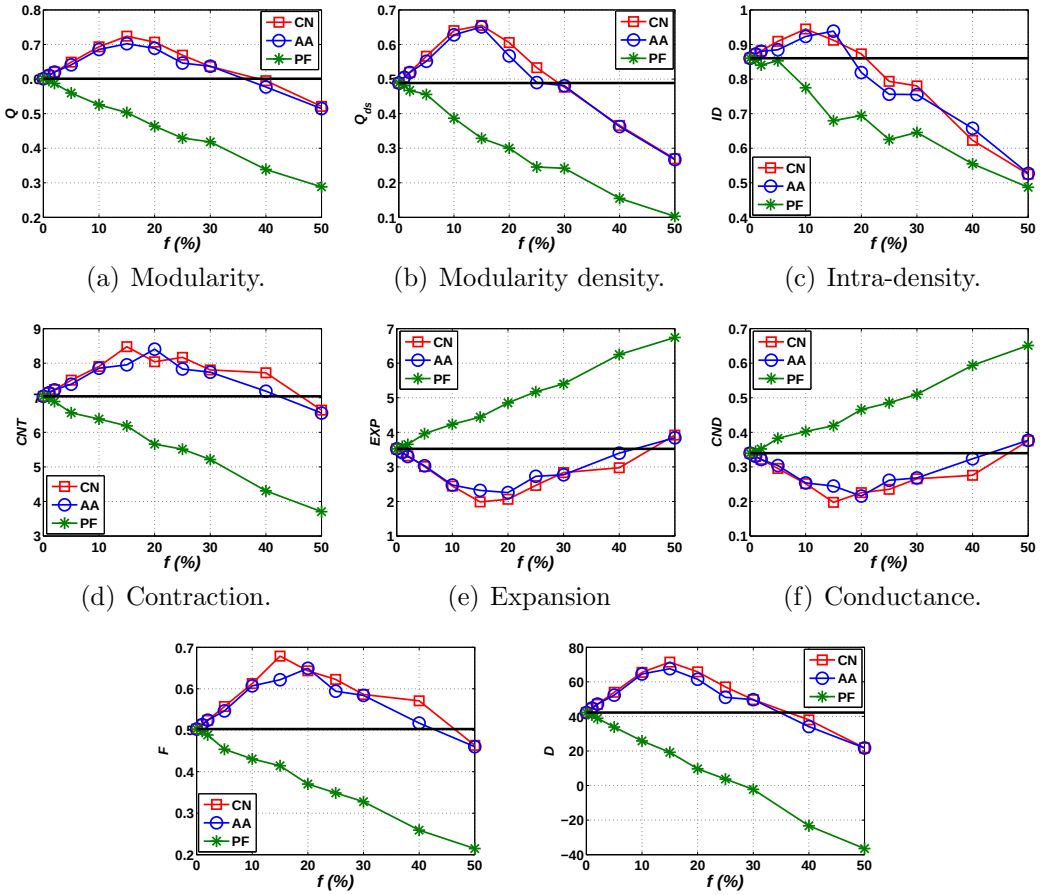
(d) Contraction. (e) Expansion (f) Conductance.

(g) Fitness. (h) Average modularity degree.

Figure 6.3: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from DBLP using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$.

the quality according to the other six metrics. With common neighbors, modularity density, intra-density, and contraction generally decrease as the replacing fraction f increases, while modularity achieves its maximum at $f = 30$, expansion and fitness function reach their minimum or maximum at $f = 40$, and conductance and D attain their minimum or maximum at $f = 15$. With Adamic-Adar, modularity, expansion, conductance, fitness function reach their maxima or minima at $f = 20$, and modularity density and average modularity degree achieve their maxima at $f = 15$. Thus, there is a limit of how many links could be replaced on DBLP. We can also notice that common neighbors metric generally performs best, followed by Adamic-Adar.

Figure 6.4 shows the results of Football dataset. From the figure, we could



(a) Modularity. (b) Modularity density. (c) Intra-density.

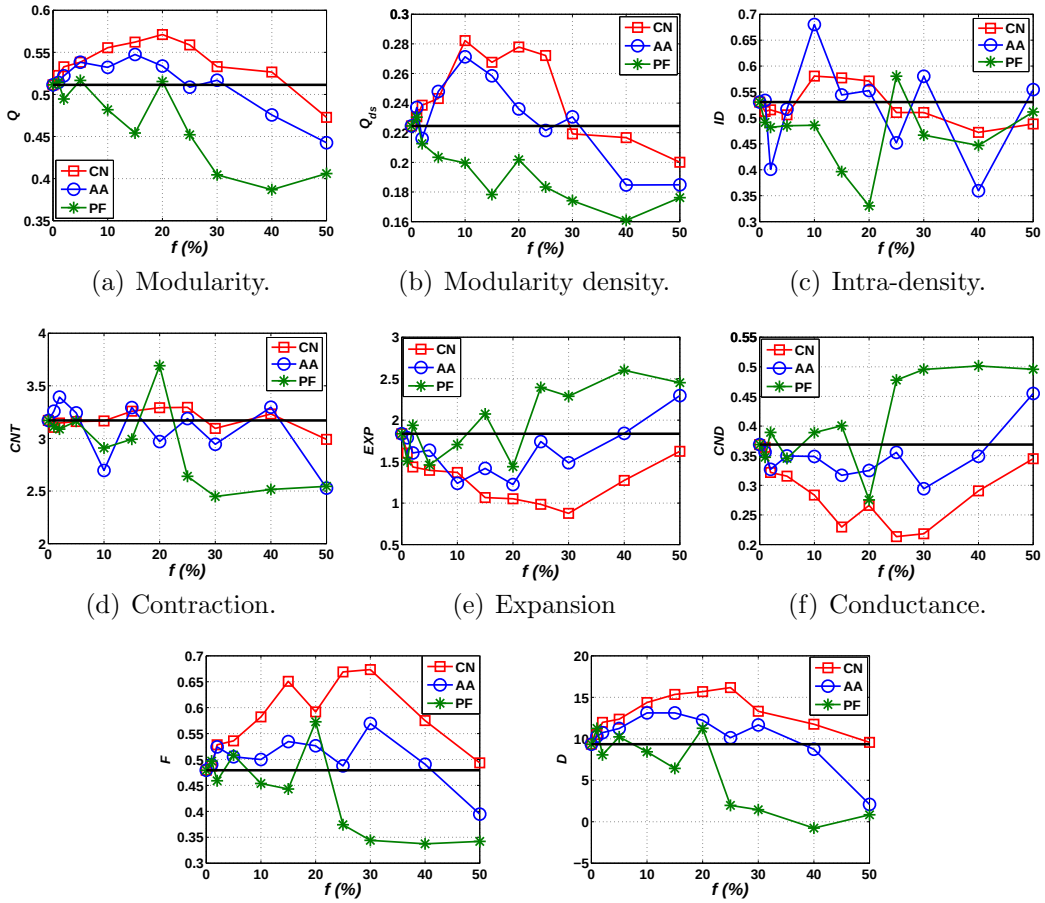
(d) Contraction. (e) Expansion (f) Conductance.

(g) Fitness. (h) Average modularity degree.

Figure 6.4: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from Football using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$**.**

observe that PropFlow has a poor performance on this dataset. As the replacing fraction f increases, the quality of community structure declines linearly. In contrast, with common neighbors and Adamic-Adar, our link improvement approach is able to significantly improve the quality of community structure. All the quality metrics get their maxima or minima at $f = 15$ or $f = 20$. Still, there is a limit of how many links could be replaced on Football dataset. We can also notice that common neighbors metric performs slightly better than Adamic-Adar.

Figure 6.5 presents the results of Dolphin dataset. It shows again that PropFlow does not work well on this network. As f increases, the quality of community structure becomes worse. While with common neighbors and Adamic-Adar, the commu-



(a) Modularity. (b) Modularity density. (c) Intra-density.

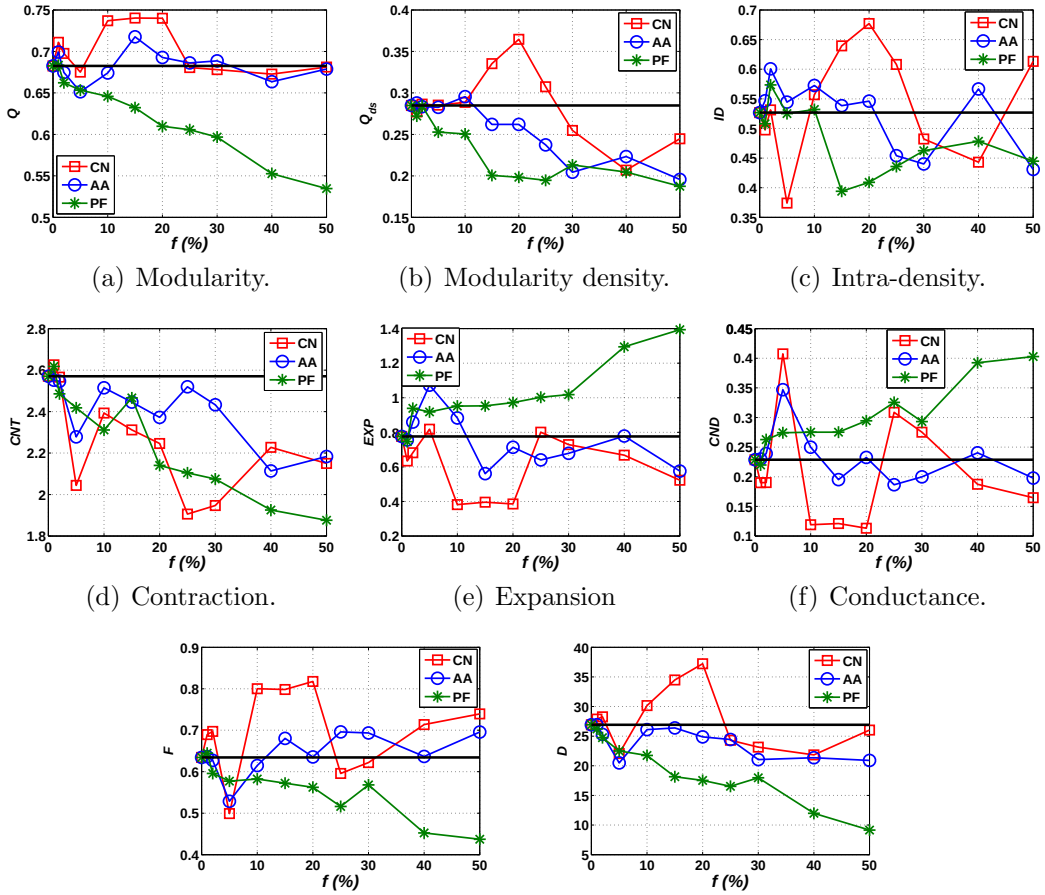
(d) Contraction. (e) Expansion (f) Conductance.

(g) Fitness. (h) Average modularity degree.

Figure 6.5: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from Dolphin using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$**.**

nity quality metric values improve until a certain value of f is reached. For instance with common neighbors, the obtained fitness function is larger than the original one for all values of f and it reaches the maximum at $f = 30$. So, there is a threshold of how many links could be replaced on Dolphin. Notice that common neighbors metric generally performs better than Adamic-Adar.

Figures 6.6 and 6.7 show the results of Santafe and Karate datasets, respectively. From the two figures, we could observe that our link improvement procedure with all the three link prediction metrics has a poor performance. All the eight community quality metrics indicate declining quality as f increases. Only with common neighbors, our link improving method could slightly improve the community quality.



(a) Modularity. (b) Modularity density. (c) Intra-density.

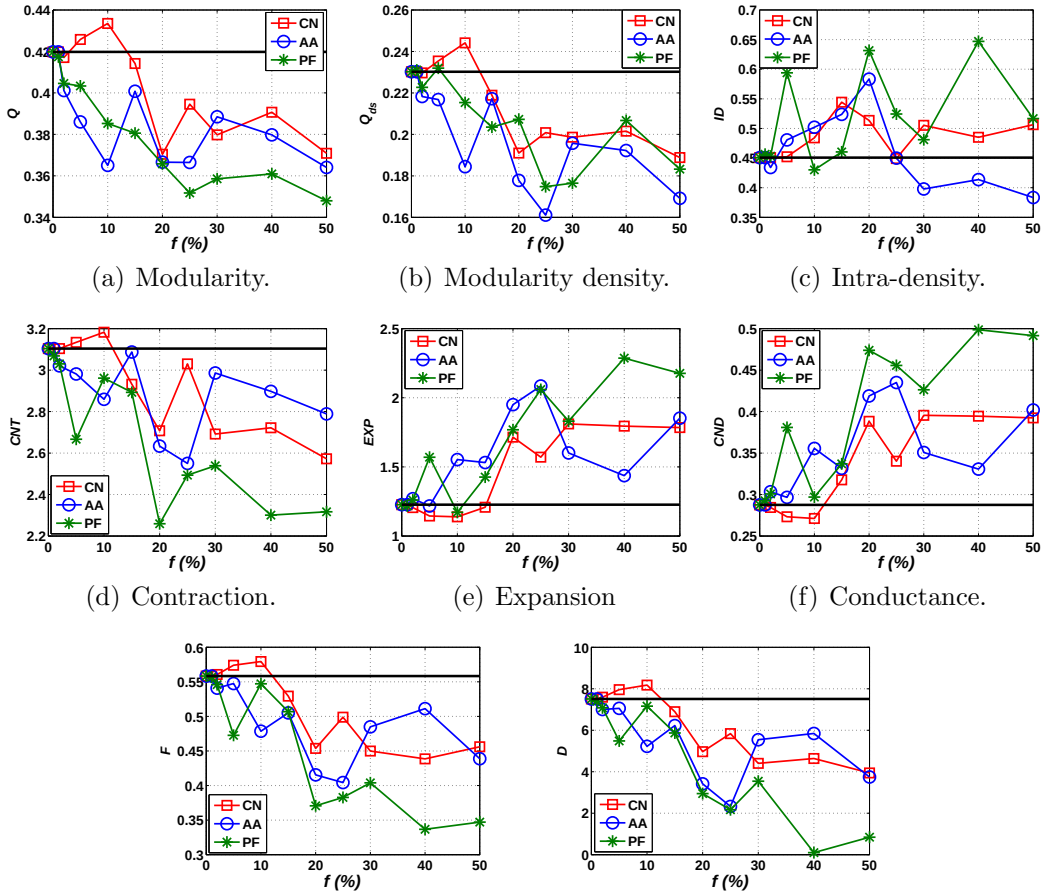
(d) Contraction. (e) Expansion (f) Conductance.

(g) Fitness. (h) Average modularity degree.

Figure 6.6: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from Santafe using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$.

On Santafe, the maxima or minima of the quality metrics are obtained at $f = 20$. On Karate, the maxima or minima are achieved at $f = 10$. The reason for the poor performance of our method is that these two networks are very small and their members were in the same club or institute for long time so they were able to evolve their links to the desired state. Thus, there is not much that can be done to improve the community structure of these two networks with our link improvement approach. In contrast, the other five networks are different, since they are either very large or their members do not have a global knowledge of the network structure or all the nodes, hence in each of these network there are many potentially good matches that are not realized even after a long time evolution [109].

We also draw a table to summarize the performance of our link improvement



(a) Modularity. (b) Modularity density. (c) Intra-density.

(d) Contraction. (e) Expansion (f) Conductance.

(g) Fitness. (h) Average modularity degree.

Figure 6.7: The community quality metric values of the community structure that SpeakEasy discovers on the networks generated from Karate using our link improvement method with $f = [0, 1, 2, 5, 10, 15, 20, 25, 30, 40, 50]$.

approach with link prediction metrics being common neighbors and Adamic-Adar on Amazon, DBLP, Football, and Dolphin datasets. We do not show the result for PropFlow because it works well only on Gowalla dataset. Also, we do not show the results for Gowalla, Santafe, and Karate since on these datasets, common neighbors or Adamic-Adar are either not the best or perform poorly. The fraction of replaced links f cell in the table shows the best f for the corresponding community quality metric. $RI = abs(best\_value - original)/orignal * 100\%$ is the relative improvement (RI) of the corresponding community quality metric attained with the best f compared to the value of the original unchanged network ($f = 0$). The ✗ in the table indicates that our link prediction method impairs the community quality according to the community quality metric in that row. This table and all

Table 6.2: The best replacing fraction f and the corresponding relative improvement (RI) of the community quality metric achieved using our link improvement method with common neighbors and Adamic-Adar on Amazon, DBLP, Football, and Dolphin.

		Datasets							
		Amazon		DBLP		Football		Dolphin	
		f	RI	f	RI	f	RI	f	RI
CN	Q	30	16.6	30	15.3	***15***	20.5	20	11.7
	Q_{ds}	25	27.8	✗	✗	***15***	34.1	10	25.7
	ID	25	12.6	✗	✗	10	9.8	10	9.4
	CNT	✗	✗	✗	✗	***15***	20.4	25	3.9
	EXP	***40***	***52.6***	***40***	***53.4***	***15***	43.6	***30***	52.3
	CND	***40***	36.0	15	13.0	***15***	41.9	25	42.1
	F	***40***	27.2	***40***	18.8	***15***	35.0	***30***	40.5
	D	25	40.8	15	18.5	***15***	***69.2***	25	***73.0***
AA	Q	***30***	8.9	***20***	10.4	15	16.9	15	7.1
	Q_{ds}	25	18.4	15	2.6	15	33.2	10	20.8
	ID	25	4.4	✗	✗	15	9.1	✗	✗
	CNT	15	3.0	✗	✗	***20***	19.4	✗	✗
	EXP	25	***28.2***	***20***	***28.8***	***20***	36.0	20	33.4
	CND	25	20.5	***20***	14.7	***20***	36.5	***30***	20.2
	F	25	15.6	***20***	13.6	***20***	29.1	***30***	18.8
	D	25	26.0	15	22.5	15	***60.1***	15	***40.4***

the above figures show that our link improvement procedure is able to significantly refine the community structure of five out of seven networks we considered.

Generally, our method has the best performance when taking common neighbors as the link prediction metric, followed by Adamic-Adar. We have also tried our link improvement method with the three link prediction metrics on three protein-protein interaction networks, however we did not get the expected results. Because of limited space, we do not show them here. We conjecture that current link prediction metrics or our link improvement approach may not be suitable for biological networks. A new approach specialized for biological networks might be necessary to improve the community structure for them. Therefore, we conclude that a single link prediction metric cannot perform well on every network, because its performance depends on the meaning of the relationships which define links in the network. Also, each link prediction metric varies its performance on different kinds of networks.

Thus, our method can be used to predict the performance of link prediction metrics. If the highly ranked predicted links do not improve quality of communities, they are unlikely to be formed quickly. We could also observe that there is a limit or threshold (which varies for different link prediction metrics) of how many links could be replaced for the purpose of improving community structure of a network. Going beyond this threshold may lead to higher cost and lower performance although the quality of the community structure may still be better than such quality for the original unchanged network.

6.4 Conclusion and Future Work

In this chapter, we introduce an approach to improve the network community structure by removing a certain fraction of low ranking existing links and replacing them with highly ranked predicted links. The ranks of the added or removed links are obtained using three link prediction metrics. The proposed method is able to significantly improve the community structure of the networks we considered. However, there is a threshold of how many links can be replaced in order to refine the community structure of a network. Going beyond this threshold may lead to higher cost and lower performance. In the experiments, we adopted three link prediction metrics, common neighbors, Adamic-Adar, and PropFlow. Generally, the link improvement method with common neighbors has the best performance, followed by Adami-Adar, while PropFlow performs extremely well only on Gowalla dataset. We argue that a single link prediction method cannot perform uniformly well on every network. Some metrics are more suitable for a particular network than others depending on the nature of the links. Finally, we noticed that there is a correlation between certain network properties and the performance of link prediction and improvement. Two influential factors that we observed are the network size and the degree to which nodes possess global knowledge about the network structure.

In the future, we plan to adopt more link prediction metrics into our link improvement approach to explore the performance of different link prediction metrics on different types of networks. We also plan to propose a new link prediction metric or a new link improvement approach specific for biological networks. In this

chapter, we only considered disjoint community structure. In the future, we plan to explore how our link improvement method could refine the quality of overlapping community structure. Moreover, we will evaluate this approach with ground truth communities on evolving networks to see whether the improved community structure is more consistent with the ground truth community structure so that we could determine whether the improvement is real. In addition, we are considering using more community detection algorithms to evaluate the performance of link prediction and improvement methods.

CHAPTER 7
CONCLUSION

In this thesis, we review the definitions of communities, the definition of modularity and its corresponding optimization approaches. Then, we discuss the two opposite yet coexisting problems of modularity maximization. To solve the two issues of modularity simultaneously, we propose a new community quality metric, called modularity density. We also extend modularity density to be able to measure the quality of overlapping community structure. We then propose a novel fine-tuned disjoint community detection algorithm that repeatedly attempts to improve the quality metrics by splitting and merging the given community structure. Finally, we introduce an link improvement approach to improve the network community structure by removing a certain fraction of low ranking existing links and replacing them with highly ranked predicted links. A summary of our findings is given below.

In Chapter 3, we introduce our newly proposed community quality metric, modularity density. It is able to simultaneously resolve the two opposite yet coexisting issues of Newman's modularity which in some cases tends to favor small communities over large ones while in others, large communities over small ones. The latter tendency is known in the literature as the resolution limit problem of modularity. We show with proofs and experiments on real-world dynamic datasets that modularity density could avoid the two issues at the same time and therefore is an effective alternative to modularity.

In Chapter 4, we review overlapping extensions of modularity and generalize them with a uniform definition enabling application of different belonging coefficients and belonging functions to select the best. We determine which versions of the belonging coefficient and the belonging function are better for measuring quality of fuzzy overlapping community structure. We find that the first version of the belonging coefficient is better than the second one, which means that the coefficient of a node belonging to a community should be the reciprocal of the number of communities to which this node belongs. In addition, we find that the second

version of the belonging function is better than the first version, meaning that the probability that two nodes belong to the same community should be the product, not the average, of their belonging coefficients. Moreover, we propose overlapping extensions for localized modularity, modularity density, and eight local community quality metrics analogous to such extension of modularity. Based on the experimental results, we recommend using the edge-based overlapping extension of modularity with the first version of belonging coefficient and with its own belonging function. We also recommend using the node-based overlapping extension of modularity and overlapping extension of modularity density with the first version of belonging coefficient and the second version of belonging function as the metrics of the global quality of overlapping community structure.

In Chapter 5, we propose an novel fine-tuned disjoint algorithm to maximize Q_{ds}. This new algorithm can actually be used to optimize any community quality metric. We evaluate the three algorithms, *Greedy* Q, *Fine-tuned* Q based on Q, and *Fine-tuned* Q_{ds} based on Q_{ds}, with seven metrics based on ground truth communities. The evaluation results imply that *Fine-tuned* Q_{ds} performs best among the three algorithms, followed by *Fine-tuned* Q. The experiments also show that *Fine-tuned* Q_{ds} can dramatically improve the community detection results of other algorithms. In addition, all the seven quality metrics based on ground truth communities are consistent with Q_{ds}, but not consistent with Q, which indicates the superiority of Q_{ds} over Q as a community quality metric.

In Chapter 6, we introduce an approach to improve the network community structure by removing a certain fraction of low ranking existing links and replacing them with highly ranked predicted links. The ranks of the added or removed links are obtained using three link prediction metrics. The proposed method is able to substantially improve the community structure of the networks we considered. However, there is a threshold of how many links can be replaced in order to refine the community structure of a network. Going beyond this threshold may lead to higher cost and lower performance. In the experiments, we adopt three link prediction metrics, common neighbors, Adamic-Adar, and PropFlow. Generally, the link improvement method with common neighbors has the best performance, followed

by Adami-Adar, while PropFlow performs extremely well only on Gowalla dataset. We argue that a single link prediction method cannot perform uniformly well on every network. Some metrics are more suitable for a particular network than others depending on the nature of the links. Finally, we notice that there is a correlation between certain network properties and the performance of link prediction and improvement. Two influential factors that we observe are the network size and the degree to which nodes possess global knowledge about the network structure.

To summarize, we propose a new community quality metric, modularity density, for measuring the quality of both disjoint and overlapping community structure, and then maximize modularity density in order to find meaningful communities. Moreover, we apply this new metric to evaluate a link improvement approach that we introduce to improve or recover the network community structure with link prediction ranking.

REFERENCES

[1] R. E. Park, *Human Communities: The City and Human Ecology.* New York, NY: Free Press, 1952.

[2] W. S. Bainbridge, "Computational sociology," in *Blackwell Encyclopedia of Sociology.* Hoboken, NJ: Blackwell Reference Online, 2007.

[3] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications.* Cambridge, U.K.: Cambridge University Press, 1994.

[4] M. Granovetter, "The strength of weak ties," *Am. J. Sociol.*, vol. 78, no. 6, pp. 1360–1380, May 1973.

[5] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, p. 026113, Feb. 2004.

[6] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, no. 3, pp. 75–174, Feb. 2010.

[7] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E*, vol. 69, no. 6, p. 066133, Jun. 2004.

[8] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Natl. Acad. Sci.*, vol. 103, no. 23, pp. 8577–8582, Feb. 2006.

[9] E. A. Leicht and M. E. J. Newman, "Community structure in directed networks," *Phys. Rev. Lett.*, vol. 100, no. 11, p. 118703, Mar. 2008.

[10] M. E. J. Newman, "Analysis of weighted networks," *Phys. Rev. E*, vol. 70, no. 5, p. 056131, Nov. 2004.

[11] S. Fortunato and M. Barthlemy, "Resolution limit in community detection," *Proc. Natl. Acad. Sci.*, vol. 104, no. 1, pp. 36–41, Jan. 2007.

[12] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: The state-of-the-art and comparative study," *ACM Comput. Surv.*, vol. 45, no. 4, pp. 43:1–43:35, Aug. 2013.

[13] S. Zhang, R.-S. Wang, and X.-S. Zhang, "Identification of overlapping community structure in complex networks using fuzzy c-means clustering," *Physica A*, vol. 374, no. 1, pp. 483–490, Jan. 2007.

[14] T. Nepusz, A. Petróczi, L. Négyessy, and F. Bazsó, "Fuzzy communities and the concept of bridgeness in complex networks," *Phys. Rev. E*, vol. 77, no. 1, p. 016107, Jan. 2008.

[15] H. Shen, X. Cheng, K. Cai, and M.-B. Hu, "Detect overlapping and hierarchical community structure in networks," *Physica A*, vol. 388, no. 8, pp. 1706–1712, Apr. 2009.

[16] H.-W. Shen, X.-Q. Cheng, and J.-F. Guo, "Quantifying and identifying the overlapping community structure in networks," *J. Stat. Mech.: Theory E.*, vol. 2009, no. 07, p. P07042, Jul. 2009.

[17] D. Chen, M. Shang, Z. Lv, and Y. Fu, "Detecting overlapping communities of weighted networks via a local algorithm," *Physica A*, vol. 389, no. 19, pp. 4177–4187, Oct. 2010.

[18] V. Nicosia, G. Mangioni, V. Carchiolo, and M. Malgeri, "Extending the definition of modularity to directed graphs with overlapping communities," *J. Stat. Mech.: Theory E.*, vol. 2009, no. 03, p. P03024, Mar. 2009.

[19] S. Gregory, "Fuzzy overlapping communities in networks," *J. Stat. Mech.: Theory E.*, vol. 2011, no. 02, p. P02017, Feb. 2011.

[20] S. Muff, F. Rao, and A. Caflisch, "Local modularity measure for network clusterizations," *Phys. Rev. E*, vol. 72, no. 5, p. 056107, Nov. 2005.

[21] M. Chen, T. Nguyen, and B. K. Szymanski, "On measuring the quality of a network community structure," in *Proc. ASE/IEEE Int. Conf. Social Computing*, Washington, DC, 2013, pp. 122–127.

[22] M. Chen, T. Nguyen, and B. K. Szymanski, "A new metric for quality of network community structure," *ASE Human J.*, vol. 2, no. 4, pp. 226–240, Sep. 2013.

[23] G. Palla, I. Dernyi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, p. 814818, Jun. 2005.

[24] S. B. Seidman, "Network structure and minimum degree," *Soc. Networks*, vol. 5, no. 3, pp. 269–287, Sep. 1983.

[25] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *Proc. Natl. Acad. Sci.*, vol. 101, no. 9, pp. 2658–2663, Jan. 2004.

[26] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee, "Self-organization and identification of web communities," *Computer*, vol. 35, no. 3, pp. 66–71, Mar. 2002.

[27] Y. Hu, H. Chen, P. Zhang, M. Li, Z. Di, and Y. Fan, "Comparative definition of community and corresponding identifying algorithm," *Phys. Rev. E*, vol. 78, no. 2, p. 026121, Aug. 2008.

[28] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, "Quantitative function for community detection," *Phys. Rev. E*, vol. 77, no. 3, p. 036109, Mar. 2008.

[29] S. Mancoridis, B. S. Mitchell, C. Rorres, Y. Chen, and E. R. Gansner, "Using automatic clustering to produce high-level system organizations of source code," in *Proc. 6th Int. Workshop Program Comprehension*, Ischia, Italy, 1998, pp. 45–52.

[30] M. Goldberg, S. Kelley, M. Magdon-Ismail, K. Mertsalov, and A. Wallace, "Finding overlapping communities in social networks," in *Proc. 2nd IEEE Int. Conf. Social Computing*, Minneapolis, MN, 2010, pp. 104–113.

[31] J. Baumes, M. K. Goldberg, M. S. Krishnamoorthy, M. Magdon-Ismail, and N. Preston, "Finding communities by clustering a graph into overlapping subgraphs," in *Proc. IADIS Int. Conf. Applied Computing*, Algarve, Portugal, 2005, pp. 97–104.

[32] S. Kelley, M. Goldberg, M. Magdon-Ismail, K. Mertsalov, and A. Wallace, "Defining and discovering communities in social networks," in *Handbook of Optimization in Complex Networks*. New York, NY: Springer US, 2012, pp. 139–168.

[33] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," in *Proc. ACM SIGKDD Workshop Mining Data Semantics*, Beijing, China, 2012, pp. 3:1–3:8.

[34] F. R. K. Chung, *Spectral Graph Theory*. Providence, RI: American Mathematical Society, 1997.

[35] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *IEEE T. Knowl. Data En.*, vol. 20, no. 2, pp. 172–188, Feb. 2008.

[36] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, no. 6, p. 066111, Dec. 2004.

[37] K. Wakita and T. Tsurumi, "Finding community structure in mega-scale social networks," in *Proc. 16th Int. Conf. World Wide Web*, Banff, Alberta, Canada, 2007, pp. 1275–1276.

[38] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech.: Theory E.*, vol. 2008, no. 10, p. P10008, Oct. 2008.

[39] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E*, vol. 74, no. 3, p. 036104, Sep. 2006.

[40] T. Richardson, P. J. Mucha, and M. A. Porter, "Spectral tripartitioning of networks," *Phys. Rev. E*, vol. 80, no. 3, p. 036111, Sep. 2009.

[41] S. White and P. Smyth, "A spectral clustering approach to finding communities in graphs," in *Proc. 2005 SIAM Int. Conf. Data Mining*, Newport Beach, CA, 2005, pp. 274–285.

[42] J. Ruan and W. Zhang, "An efficient spectral algorithm for network community discovery and its applications to biological and social networks," in *Proc. 7th IEEE Int. Conf. Data Mining*, Omaha, NE, 2007, pp. 643–648.

[43] J. Ruan and W. Zhang, "Identifying network communities with a high resolution," *Phys. Rev. E*, vol. 77, no. 1, p. 016104, Jan. 2008.

[44] M. E. J. Newman, "Spectral methods for network community detection and graph partitioning," *Phys. Rev. E*, vol. 88, no. 4, p. 042822, Oct. 2013.

[45] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Phys. Rev. E*, vol. 72, no. 2, p. 027104, Aug. 2005.

[46] G. R and N. A. LA, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, no. 7028, pp. 895–900, Feb. 2005.

[47] R. Guimerà and L. A. N. Amaral, "Cartography of complex networks: modules and universal roles," *J. Stat. Mech.: Theory E.*, vol. 2005, no. 02, p. P02001, Feb. 2005.

[48] C. P. Massen and J. P. K. Doye, "Identifying communities within energy landscapes," *Phys. Rev. E*, vol. 71, no. 4, p. 046101, Apr. 2005.

[49] A. Medus, G. Acuña, and C. Dorso, "Detection of community structures in networks via global optimization," *Physica A*, vol. 358, no. 24, pp. 593–604, Dec. 2005.

[50] M. Sales-Pardo, R. Guimerà, A. A. Moreira, and L. a. N. Amaral, "Extracting the hierarchical organization of complex systems," *Proc. Natl. Acad. Sci.*, vol. 104, no. 39, pp. 15 224–9, Sep. 2007.

[51] G. Agarwal and D. Kempe, "Modularity-maximizing graph communities via mathematical programming," *Eur. Phys. J. B*, vol. 66, no. 3, pp. 409–418, Nov. 2008.

[52] T. Chakraborty, S. Srinivasan, N. Ganguly, S. Bhowmick, and A. Mukherjee, "Constant communities in complex networks," *Sci. Rep.*, vol. 3, p. 1825, May 2013.

[53] J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2000.

[54] S. Boettcher and A. G. Percus, "Optimization with extremal dynamics," *Phys. Rev. Lett.*, vol. 86, no. 23, pp. 5211–5214, Jun. 2001.

[55] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

[56] H. Karloff, *Linear Programming*. Cambridge, MA: Birkhäuser Boston, 1991.

[57] J. W. Berry, B. Hendrickson, R. A. LaViolette, and C. A. Phillips, "Tolerating the community detection resolution limit with edge weighting," *Phys. Rev. E*, vol. 83, no. 5, p. 056119, May 2011.

[58] A. Arenas, A. Fernandez, and S. Gomez, "Analysis of the structure of complex networks at different resolution levels," *New J. Phys.*, vol. 10, no. 5, p. 053039, May 2008.

[59] A. Lancichinetti and S. Fortunato, "Limits of modularity maximization in community detection," *Phys. Rev. E*, vol. 84, no. 6, p. 066122, Dec. 2011.

[60] C. Granell, S. Gomez, and A. Arenas, "Hierarchical multiresolution method to overcome the resolution limit in complex networks," *Int. J. Bifurcat. Chaos*, vol. 22, no. 07, p. 1250171, Jan. 2012.

[61] V. A. Traag, P. Van Dooren, and Y. Nesterov, "Narrow scope for resolution-limit-free community detection," *Phys. Rev. E*, vol. 84, no. 1, p. 016114, Jul. 2011.

[62] J. Reichardt and S. Bornholdt, "Partitioning and modularity of graphs with arbitrary degree distribution," *Phys. Rev. E*, vol. 76, no. 1, p. 015102, Jul. 2007.

[63] B. H. Good, Y.-A. de Montjoye, and A. Clauset, "Performance of modularity maximization in practical contexts," *Phys. Rev. E*, vol. 81, no. 4, p. 046106, Apr. 2010.

[64] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Phys. Rev. E*, vol. 74, no. 1, p. 016110, Jul. 2006.

[65] V. Kawadia and S. Sreenivasan, "Sequential detection of temporal communities by estrangement confinement," *Sci. Rep.*, vol. 2, p. 794, Nov. 2012.

[66] A. S. Waugh, L. Pei, J. H. Fowler, P. J. Mucha, and M. A. Porter, "Party polarization in congress: A network science approach," *arXiv preprint arXiv:0907.3509 [physics.soc-ph]*, (Accessed Sep. 01, 2013).

[67] N. Eagle, A. Pentland, and D. Lazer, "Inferring social network structure using mobile phone data," *Proc. Natl. Acad. Sci.*, vol. 106, no. 36, pp. 15 274–15 278, Sep. 2009.

[68] J. Xie, M. Chen, and B. K. Szymanski, "LabelrankT: Incremental community detection in dynamic networks via label propagation," in *Proc. ACM SIGMOD Workshop Dynamic Networks Management and Mining*, New York, NY, 2013, pp. 25–32.

[69] M. Chen, S. Liu, and B. K. Szymanski, "Parallel toolkit for measuring the quality of network community structure," in *Proc. 2014 European Network Intelligence Conf.*, Wroclaw, Poland, 2014, pp. 22–29.

[70] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New J. Phys.*, vol. 11, no. 3, p. 033015, Mar. 2009.

[71] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, "Quantitative function for community detection," *Phys. Rev. E*, vol. 77, no. 3, p. 036109, Mar. 2008.

[72] J. Xie and B. K. Szymanski, "Towards linear time overlapping community detection in social networks," in *16th Pacific-Asia Conf. Knowledge Discovery and Data Mining*, Kuala Lumpur, Malaysia, 2012, pp. 25–36.

[73] K. Kuzmin, M. Chen, and B. K. Szymanski, "Parallelizing SLPA for scalable overlapping community detection," *Sci. Program.*, vol. 2015, no. 1, p. 461362, Mar. 2015.

[74] C. Gaiteri, M. Chen, B. K. Szymanski, K. Kuzmin, J. Xie, C. Lee, T. Blanche, E. C. Neto, S.-C. Huang, T. Grabowski, T. Madhyastha, and V. Komashko, "Identifying robust communities and multi-community nodes by combining top-down and bottom-up approaches to clustering," *Sci. Rep.*, vol. 5, p. 16361, Nov. 2015.

[75] S. Gregory, "Finding overlapping communities in networks by label propagation," *New J. Phys.*, vol. 12, no. 10, p. 103018, Oct. 2010.

[76] D. Lusseau, K. Schneider, O. Boisseau, P. Haase, E. Slooten, and S. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behav. Ecol. Sociobiol.*, vol. 54, no. 4, pp. 396–405, Jun. 2003.

[77] R. Guimerà, L. Danon, A. DíazGuilera, F. Giralt, and A. Arenas, "Self-similar community structure in a network of human interactions," *Phys. Rev. E*, vol. 68, no. 6, p. 065103, Dec. 2003.

[78] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Natl. Acad. Sci.*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002.

[79] P. Gleiser and L. Danon, "Community structure in jazz," *Adv. Complex Syst.*, vol. 06, no. 04, pp. 565–573, Dec. 2003.

[80] W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropol. Res.*, vol. 33, no. 4, pp. 452–473, Dec. 1977.

[81] D. E. Knuth, *The Stanford GraphBase: A Platform for Combinatorial Computing*. Reading, MA: Addison-Wesley, 1993.

[82] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, "Models of social networks based on social distance attachment," *Phys. Rev. E*, vol. 70, no. 5, p. 056122, Nov. 2004.

[83] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 u.s. election: Divided they blog," in *Proc. 3rd Int. Workshop Link Discovery*, Chicago, IL, 2005, pp. 36–43.

[84] V. Krebs. (2004) Books about us politics. [Online]. Available: http://www.orgnet.com/ (Accessed Jul. 01, 2015).

[85] T. Chakraborty, S. Srinivasan, N. Ganguly, A. Mukherjee, and S. Bhowmick, "On the permanence of vertices in network communities," in *Proc. 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, New York, NY, 2014, pp. 1396–1405.

[86] S. R. Collins, K. P. Kemmeren, F. X. chu Zhao, G. J. F. Greenblatt, F. Spencer *et al.*, "Toward a comprehensive atlas of the physical interactome of saccharomyces cerevisiae," *Mol. Cell. Proteomic*, vol. 6, no. 3, pp. 439–450, Jan. 2007.

[87] A.-C. Gavin, P. Aloy, P. Grandi *et al.*, "Proteome survey reveals modularity of the yeast cell machinery," *Nature*, vol. 440, no. 7084, pp. 631–636, Mar. 2006.

[88] H. W. Mewes, C. Amid, R. Arnold *et al.*, "MIPS: analysis and annotation of proteins from whole genomes," *Nucleic Acids Res.*, vol. 32, pp. D41–D44, Jan. 2004.

[89] E. L. Hong, R. Balakrishnan, Q. Dong *et al.*, "Gene ontology annotations at SGD: new data sources and annotation methods," *Nucleic Acids Res.*, vol. 36, pp. D577–D581, Jan. 2008.

[90] S. Pu, J. Wong, B. Turner, E. Cho, and S. J. Wodak, "Up-to-date catalogues of yeast protein complexes," *Nucleic Acids Res.*, vol. 37, no. 3, pp. 825–831, Feb. 2009.

[91] M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE T. Comput. Soc. Syst.*, vol. 1, no. 1, pp. 46–65, Mar. 2014.

[92] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Phys. Rev. E*, vol. 80, no. 1, p. 016118, Jul. 2009.

[93] Y.-C. Wei and C.-K. Cheng, "Towards efficient hierarchical designs by ratio cut partitioning," in *Proc. IEEE Int. Conf. Computer-Aided Design*, Santa Clara, CA, 1989, pp. 298–301.

[94] M. Fiedler, "Algebraic connectivity of graphs," *Czech. Math. J.*, vol. 23, no. 2, pp. 298–305, Jan. 1973.

[95] A. Pothen, H. D. Simon, and K.-P. Liou, "Partitioning sparse matrices with eigenvectors of graphs," *SIAM J. Matrix Anal. A.*, vol. 11, no. 3, pp. 430–452, May 1990.

[96] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *J. Stat. Mech.: Theory E.*, vol. 2005, no. 09, p. P09008, Sep. 2005.

[97] S. Wagner and D. Wagner, "Comparing Clusterings – An Overview," Universität Karlsruhe (TH), Karlsruhe, Germany, Tech. Rep. 2006-04, 2007.

[98] S. Van Dongen, "Performance criteria for graph clustering and markov cluster experiments," Nat. Res. Inst. Math. Comput. Sci., Amsterdam, The Netherlands, The Netherlands, Tech. Rep. INS-R0012, May 2000.

[99] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Am. Stat. Assoc.*, vol. 66, no. 336, pp. 846–850, Dec. 1971.

[100] L. Hubert and P. Arabie, "Comparing partitions," *J. Classif.*, vol. 2, no. 1, pp. 193–218, Dec. 1985.

[101] A. Ben-Hur, A. Elisseeff, and I. Guyon, "A stability based method for discovering structure in clustered data," in *Proc. Pacific Symp. Biocomputing*, Kauai, HI, 2002, pp. 6–17.

[102] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," in *Proc. Int. AAAI Conf. Weblogs and Social Media*, San Jose, CA, 2009, pp. 361–362.

[103] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, no. 4, p. 046110, Oct. 2008.

[104] L. Adamic and E. Adar, "Friends and neighbors on the web," *Soc. Networks*, vol. 25, no. 3, pp. 211–230, Jan. 2003.

[105] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *Proc. 16th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Washington, DC, 2010, pp. 243–252.

[106] M. Chen, K. Kuzmin, and B. K. Szymanski, "Extension of modularity density for overlapping community structure," in *Proc. 2014 IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining*, Beijing, China, 2014, pp. 856–863.

[107] M. Chen and B. K. Szymanski, "Fuzzy overlapping community quality metrics," *Soc. Netw. Anal. Min.*, vol. 5, no. 1, pp. 1–14, Jul. 2015.

[108] T. Nguyen, M. Chen, and B. K. Szymanski, "Analyzing the proximity and interactions of friends in communities in Gowalla," in *Proc. IEEE 13th Int. Conf. Data Mining Workshops*, Dallas, TX, 2013, pp. 1036–1044.

[109] T. Jia, R. Spivey, B. K. Szymanski, and G. Korniss, "An analysis of the matching hypothesis in networks," *PLoS One*, vol. 10, no. 6, p. e0129804, Jun. 2015.

[110] B. Yan and S. Gregory, "Finding missing edges and communities in incomplete networks," *J. Phys. A*, vol. 44, no. 49, p. 495102, Nov. 2011.

[111] B. Yan and S. Gregory, "Finding missing edges in networks based on their community structure," *Phys. Rev. E*, vol. 85, no. 5, p. 056112, May 2012.

[112] Z. Liu, J.-L. He, K. Kapoor, and J. Srivastava, "Correlations between community structure and link formation in complex networks," *PLoS ONE*, vol. 8, no. 9, p. 72908, Sep. 2013.

[113] Z. Liu, W. Dong, and Y. Fu, "Local degree blocking model for missing link prediction in complex networks," *Chaos*, vol. 5, no. 1, p. 013115, Jan. 2015.

[114] B. Yan and S. Gregory, "Detecting community structure in networks using edge prediction methods," *J. Stat. Mech.: Theory E.*, vol. 2012, no. 09, p. P09008, Sep. 2012.

[115] M. Burgess, E. Adar, and M. Cafarella, "Link-prediction enhanced consensus clustering for complex networks," *arXiv preprint arXiv:1506.01461 [cs.SI]*, (Accessed Jul. 01, 2015).

[116] R. N. Lichtenwalter and N. V. Chawla, "Lpmade: Link prediction made easy," *J. Mach. Learn. Res.*, vol. 12, pp. 2489–2492, Nov. 2011.

[117] L. L and T. Zhou, "Link prediction in complex networks: A survey," *Physica A*, vol. 390, no. 6, pp. 1150–1170, Mar. 2011.

[118] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Am. Soc. Inf. Sci. Tec.*, vol. 58, no. 7, pp. 1019–1031, May 2007.

[119] A. Trias Mansilla, M. Chen, B. K. Szymanski, and J. de la Rosa Esteva, "Naming game dynamics on pairs of connected networks with competing opinions," in *Social Informatics*. Barcelona, Spain: Springer International Publishing, 2015, vol. 8852, pp. 368–379.

[120] I. Farkas, D. bel, G. Palla, and T. Vicsek, "Weighted network modules," *New J. Phys.*, vol. 9, no. 6, p. 180, Jun. 2007.

APPENDIX A
APPENDIX OF CHAPTER 4

A.1 Proof of the Equivalence between Q_{ov} and Q'_{ov}

Here, we prove that the second term of Q_{ov}, Equation (4.16), is equal to the second term of Q'_{ov}, Equation (4.17).

$$\begin{aligned}
\left(2|E_c^{in}| + |E_c^{out}|\right)^2 &= \left(\sum_{i,j \in c} a_{i,c} a_{j,c} A_{ij} + \sum_{i \in c} \sum_{\substack{c' \in C \\ c' \neq c \\ j \in c'}} a_{i,c} a_{j,c'} A_{ij} \right)^2 \\
&= \left(\sum_{i \in c} a_{i,c} \sum_{j \in c} a_{j,c} A_{ij} + \sum_{i \in c} a_{i,c} \sum_{\substack{c' \in C \\ c' \neq c \\ j \in c'}} a_{j,c'} A_{ij} \right)^2 \\
&= \left(\sum_{i \in c} a_{i,c} \sum_{j \in V} A_{ij} \right)^2 \\
&= \left(\sum_{i \in c} k_i a_{i,c} \right)^2 \\
&= \sum_{i \in c} k_i a_{i,c} \sum_{j \in c} k_j a_{j,c} \\
&= \sum_{i,j \in c} k_i k_j a_{i,c} a_{j,c}.
\end{aligned} \tag{A.1}$$

A.2 Real-world Network Datasets

A.2.1 C. elegans Metabolic Network

This is the metabolic network of C. elegans [45] with 453 nodes and 2025 edges. Table A.1 shows the best value of threshold r for SLPA [72, 73], the best value of parameter k for CFinder [23, 120], and the best value of threshold tr for SpeakEasy [74] determined by the twelve community quality metrics with four possible com-

Portions of this chapter previously appeared as: M. Chen and B. K. Szymanski, "Fuzzy overlapping community quality metrics," *Soc. Netw. Anal. Min.*, vol. 5, no. 1, pp. 1-14, Jul. 2015.

Table A.1: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on C. elegans metabolic network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.3	0.3	0.35	0.05	0.5	0.05	0.5	0.5	0.5	0.05	0.5	***0.5 (6)***
	(1,2)	0.3	0.3	0.3	0.4	0.05	0.5	0.05	0.5	0.4	0.05	0.05	0.4	0.05 (4)
	(2,1)	0.5	0.3	0.45	0.4	0.05	0.5	0.05	0.5	0.5	0.5	0.05	0.5	***0.5 (6)***
	(2,2)	0.4	0.4	0.45	0.4	0.05	0.35	0.05	0.35	0.4	0.05	0.05	0.4	0.4 (5)
CFinder	(1,1)	9	3	4	3	3	4	4	7	7	3	3	3	3 (6)
	(1,2)	4	3	4	4	3	4	3	7	3	3	3	3	***3 (7)***
	(2,1)	7	3	4	3	3	4	4	9	9	3	3	9	3 (5)
	(2,2)	5	9	4	4	3	4	3	9	9	9	3	3	{3,9} (4)
SpeakEasy	(1,1)	0.75	0.8	0.75	0.75	0.05	0.8	0.05	1	0.75	0.9	0.75	0.75	0.75 (6)
	(1,2)	0.75	0.8	0.75	0.75	0.75	0.8	0.9	0.35	0.75	0.9	0.75	0.75	***0.75 (7)***
	(2,1)	0.75	0.8	0.75	0.75	0.05	0.8	0.05	1	0.75	0.9	0.75	0.75	0.75 (6)
	(2,2)	0.75	0.8	0.75	0.75	0.75	0.8	0.9	0.35	0.75	0.5	0.9	0.75	0.75 (6)

binations of the two versions of belonging coefficient and two version of belonging function for this network. The last column in this table (and all the following tables) is the best value of threshold r for SLPA, the best value of parameter k for CFinder, or the best value of the threshold tr for SpeakEasy along with the corresponding number of community quality metrics (out of twelve) that are consistent with each other on determining this best r, this best k, and this best tr for each combination of belonging coefficient and belonging function. The table shows that the first version of belonging function is better than the second version of belonging function when using SLPA. For CFinder and SpeakEasy, it implies that (BC,BF)=(1,2) is the best among the four possible combinations of two versions of belonging coefficient and two versions of belonging function. In conclusion, two out of three algorithms show that (BC,BF)=(1,2) is the best on C. elegans metabolic network.

A.2.2 Dolphin Social Network

This is a social network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand [76]. There are 62 nodes and 159 edges. Table A.2 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two

Table A.2: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on dolphin social network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.4	0.4	0.4	0.05	0.5	0.05	0.5	0.5	0.45	0.5	0.45	0.5 (5)
	(1,2)	0.4	0.4	0.4	0.4	0.05	0.4	0.05	0.5	0.4	0.4	0.05	0.4	***0.4 (8)***
	(2,1)	0.5	0.4	0.05	0.4	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.45	0.5 (6)
	(2,2)	0.4	0.4	0.05	0.4	0.05	0.3	0.05	0.3	0.4	0.05	0.05	0.4	{0.05,0.4} (5)
CFinder	(1,1)	3	3	3	3	3	3	3	4	3	3	3	3	3 (11)
	(1,2)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(2,1)	3	3	3	4	3	3	3	5	5	5	3	4	3 (7)
	(2,2)	3	3	3	4	3	3	3	5	5	5	3	4	3 (7)
SpeakEasy	(1,1)	0.4	0.55	0.4	0.2	0.15	0.85	0.05	0.7	1	0.45	0.45	0.4	0.4 (3)
	(1,2)	0.4	0.2	0.4	0.2	0.4	0.85	0.45	0.7	1	0.45	0.45	0.4	***0.4 (4)***
	(2,1)	0.15	0.8	0.15	0.8	0.15	0.85	0.15	0.7	0.7	0.45	0.45	0.7	***0.15 (4)***
	(2,2)	0.4	0.8	0.15	0.8	0.4	0.85	0.45	0.7	0.7	0.7	0.45	0.7	***0.7 (4)***

versions of belonging coefficient and two version of belonging function on dolphin social network. We could learn from the table that all the three algorithms demonstrate that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.3 Email network

This network represents email interchanges between members of the Univeristy Rovira i Virgili (Tarragona) [77]. It has 1133 nodes and 5451 edges. Table A.3 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on email network. It can be observed from this table that all three algorithms demonstrate that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.4 American College Football Network

The network represents the schedule of games between college football teams in a single season [78]. There are 115 nodes and 613 edges. Table A.4 shows the best

Table A.3: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on email network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.5	0.5	0.35	0.1	0.4	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(1,2)	0.5	0.5	0.5	0.5	0.5	0.4	0.5	0.35	0.5	0.5	0.5	0.5	***0.5 (10)***
	(2,1)	0.5	0.5	0.3	0.4	0.1	0.4	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
	(2,2)	0.5	0.5	0.3	0.5	0.5	0.4	0.5	0.35	0.4	0.5	0.5	0.5	0.5 (8)
CFinder	(1,1)	4	3	3	3	3	3	4	3	3	3	3	3	***3 (10)***
	(1,2)	4	3	3	3	3	3	4	3	3	3	3	3	***3 (10)***
	(2,1)	4	5	3	4	3	3	4	10-12	9	10-12	4	8	4 (4)
	(2,2)	4	7	3	5	3	3	4	9-12	9-12	9-12	4	7	{3,4,9-12} (3)
SpeakEasy	(1,1)	0.3	0.85	1	0.8	0.05	0.75	0.05	0.7	0.55	0.9	0.9	0.9	***0.9 (3)***
	(1,2)	1	0.85	1	0.8	1	0.75	0.85	0.7	0.5	0.9	0.9	0.5	***1 (3)***
	(2,1)	0.3	0.85	1	0.8	0.05	0.75	0.05	0.7	0.95	0.9	0.9	0.5	{0.05,0.9} (2)
	(2,2)	1	0.85	1	0.8	1	0.75	0.85	0.7	0.5	0.5	0.9	0.5	***{0.5,1} (3)***

Table A.4: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on American college football network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.4	0.45,0.5	0.45,0.5	0.45,0.5	0.05	0.45,0.5	0.05	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	{0.45,0.5} (9)
	(1,2)	0.4	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	***{0.45,0.5} (11)***
	(2,1)	0.4	0.45,0.5	0.3	0.45,0.5	0.05	0.45,0.5	0.05	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	0.45,0.5	{0.45,0.5} (8)
	(2,2)	0.4	0.45,0.5	0.3	0.45,0.5	0.25	0.45,0.5	0.25	0.25	0.25	0.25	0.25	0.25	0.25 (7)
CFinder	(1,1)	4	5	4	4	3	4	3	9	4	3	3	4	***4 (6)***
	(1,2)	4	5	4	4	3	4	3	9	3	3	3	4	{3,4} (5)
	(2,1)	4	6	4	4	3	4	3	9	9	9	3	4	4 (5)
	(2,2)	4	6	4	4	3	4	3	9	9	9	3	4	4 (5)
SpeakEasy	(1,1)	0.1,0.6	0.75	0.6	0.1,0.6	0.1,0.6	0.75	0.1,0.6	0.1,0.6	0.1,0.6	0.1,0.6	0.6	0.1,0.6	***0.6 (10)***
	(1,2)	0.1,0.6	0.75	0.6	0.1,0.6	0.1,0.6	0.75	0.1,0.6	0.1,0.6	0.1,0.6	0.1,0.6	0.6	0.1,0.6	***0.6 (10)***
	(2,1)	0.1,0.6	0.75	0.6	0.1,0.6	0.1,0.6	0.75	0.1,0.6	0.1,0.6	0.1,0.6	0.1,0.6	0.6	0.1,0.6	***0.6 (10)***
	(2,1)	0.1,0.6	0.75	0.6	0.1,0.6	0.1,0.6	0.75	0.1,0.6	0.1,0.6	0.1,0.6	0.1,0.6	0.6	0.1,0.6	***0.6 (10)***

value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on American college football network. It can be observed from this table that SLPA performs best with (BC,BF)=(1,2), CFinder implies that (BC,BF)=(1,1) is the best, while SpeakEasy has no preferences.

Table A.5: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on jazz musicians network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.5	0.5	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (10)***
	(1,2)	0.5	0.5	0.5	0.5	0.1	0.5	0.1	0.5	0.5	0.5	0.5	0.5	***0.5 (10)***
	(2,1)	0.5	0.5	0.4	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
	(2,2)	0.5	0.5	0.4	0.5	0.1	0.5	0.1	0.5	0.5	0.5	0.5	0.5	0.5 (9)
CFinder	(1,1)	14	3	10	10	3	8	3	3	3	3	3	3	***3 (8)***
	(1,2)	10	3	10	10	3	8	3	3	3	3	3	3	***3 (8)***
	(2,1)	14	10	8	10	3	8	3	3	3	18	3	3	3 (6)
	(2,2)	10	17	8	10	3	8	3	3	19,20	19,20	3	3	3 (5)
SpeakEasy	(1,1)	0.75	0.5	0.75	0.75	0.1	0.85	0.1	0.75	0.75	0.55	0.55	0.8	0.75 (5)
	(1,2)	0.75	0.5	0.75	0.75	0.75	0.85	0.7	0.75	0.5	0.55	0.55	0.5	0.75 (5)
	(2,1)	0.75	0.5	0.75	0.75	0.1	0.85	0.1	0.75	0.75	0.55	0.55	0.8	0.75 (5)
	(2,2)	0.75	0.5	0.75	0.75	0.75	0.85	0.7	0.75	0.75	0.8	0.55	0.8	***0.75 (6)***

A.2.5 Jazz Musicians Network

This is a network with 198 nodes and 2742 edges of collaborations between jazz musicians [79]. Table A.5 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on jazz musicians network. From this table, we could see that the first version of belonging coefficient is better than the second version when using SLPA and CFinder, while SpeakEasy demonstrates that (BC,BF)=(2,2) is the best among the four combinations. In summary, two of the three algorithms support that the first version of belonging coefficient is better than the second one on jazz musicians network.

A.2.6 Zachary's Karate Club Network

This network represents the friendships between 34 members of a karate club at a US university during two years [80]. It has 34 nodes and 78 edges. Table A.6 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of

Table A.6: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Zachary's karate club network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.5	0.45	0.5	0.1	0.35	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(1,2)	0.45	0.45	0.45	0.45	0.15	0.35	0.45	0.45	0.45	0.45	0.45	0.45	***0.45 (10)***
	(2,1)	0.5	0.5	0.45	0.5	0.1	0.35	0.05	0.5	0.5	0.5	0.5	0.45	0.5 (7)
	(2,2)	0.45	0.45	0.45	0.45	0.15	0.35	0.45	0.45	0.45	0.45	0.45	0.45	***0.45 (10)***
CFinder	(1,1)	3	3	3	3	3	3	3	5	3	3	3	3	***3 (11)***
	(1,2)	3	3	3	3	3	3	3	5	3	3	3	3	***3 (11)***
	(2,1)	4	3	3	3	3	3	3	5	5	5	3	3	3 (8)
	(2,2)	4	4	3	3	3	3	3	5	5	5	3	3	3 (7)
SpeakEasy	(1,1)	0.45	0.45	0.45	0.45	0.2	0.95	0.05	0.7,0.75	0.45	0.65	0.65	0.65	0.45 (5)
	(1,2)	0.45	0.45	0.45	0.45	0.45	0.95	0.65	0.15	0.45	0.65	0.65	0.65	***0.45 (6)***
	(2,1)	0.45	0.45	0.15	0.45	0.2	0.95	0.05	0.9	0.45	0.65	0.65	0.45	0.45 (5)
	(2,2)	0.45	0.45	0.15	0.45	0.2	0.35	0.65	0.9	0.45	0.45	0.85	0.45	***0.45 (6)***

Table A.7: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Les Miserables network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.35	0.5	0.25	0.35	0.05	0.5	0.05	0.5	0.5	0.25	0.25	0.25	{0.25,0.5} (4)
	(1,2)	0.35	0.5	0.25	0.35	0.15	0.5	0.25	0.35	0.25	0.25	0.25	0.25	***0.25 (6)***
	(2,1)	0.35	0.5	0.15	0.4	0.05	0.5	0.05	0.5	0.5	0.25	0.25	0.25	0.5 (4)
	(2,2)	0.35	0.5	0.15	0.35	0.15	0.5	0.15	0.35	0.25	0.15	0.15	0.25	0.15 (5)
CFinder	(1,1)	6	3	4	4	3	3	5	8	3	3	3	3	3 (7)
	(1,2)	5	3	4	5	3	3	3	4	3	3	3	3	***3 (8)***
	(2,1)	6	6	4	6	3	3	5	9	9	9	3	6	6 (4)
	(2,2)	6	6	4	6	3	3	3	9	9	9	3	6	{3,6} (4)
SpeakEasy	(1,1)	0.65	0.85	0.65	0.6	0.85	0.85	0.1	0.55,0.95	0.55	0.85	0.85	0.6	0.85 (5)
	(1,2)	0.65	0.85	0.65	0.6	0.85	0.85	0.85	0.55,0.95	0.55	0.85	0.85	0.6	***0.85 (6)***
	(2,1)	0.65	0.7	0.65,0.8	0.6	0.85	0.85	0.1	0.95	0.55	0.85	0.85	0.6	0.85 (4)
	(2,2)	0.65,0.8	0.7	0.65,0.8	0.45,0.6	0.85	0.85	0.85	0.45,0.95	0.55	0.95	0.85	0.45,0.6	0.85 (4)

belonging coefficient and two version of belonging function on Zachary's karate club network. It can be observed from this table that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.7 Les Miserables Network

This is a coappearance network of characters in the novel Les Miserables [81]. It has 77 nodes and 254 edges. Table A.7 shows the best value of threshold r for

Table A.8: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Network Science coauthorship network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.4	0.5	0.35	0.05	0.4	0.05	0.5	0.5	0.35	0.35	0.35	0.35 (4)
	(1,2)	0.5	0.4	0.5	0.4	0.5	0.4	0.25	0.5	0.35	0.2	0.15	0.35	0.5 (4)
	(2,1)	0.05	0.4	0.35	0.35	0.05	0.4	0.05	0.5	0.5	0.35	0.35	0.35	***0.35 (5)***
	(2,2)	0.5	0.4	0.35	0.35	0.15	0.4	0.25	0.5	0.5	0.2	0.15	0.35	{0.35,0.5} (3)
CFinder	(1,1)	3	3	3	3	3	3	3	7	3	3	3	3	3 (11)
	(1,2)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(2,1)	3	3	3	3	3	3	3	11-20	11-20	11-20	3	3	3 (9)
	(2,2)	3	3	3	3	3	3	3	9-20	9-20	9-20	3	3	3 (9)
SpeakEasy	(1,1)	0.05	0.25	0.25	0.25	0.05	0.7	0.05	1	1	0.7	0.7	0.7	0.7 (4)
	(1,2)	0.25	0.25	0.25	0.25	0.25	0.7	0.7	0.25	0.25	0.25	0.25	0.25	***0.25 (10)***
	(2,1)	0.05	0.3	0.15	0.25	0.05	0.7	0.05	0.85	1	0.25	0.7	1	0.05 (3)
	(2,2)	0.15	0.3	0.15	0.25	0.25	0.7	0.7	0.4	0.2	0.2	0.9	1	{0.15,0.2,0.25,0.7} (2)

SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Les Miserables network. The table shows that for all three algorithms (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.8 Network Science Coauthorship Network

This is a coauthorship network of scientists working on network theory and experiment [39]. There are 1461 nodes and 2742 edges. Table A.8 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Network Science coauthorship network. We can observe from this table that SLPA shows that (BC,BF)=(2,1) is the best among the four combinations, while CFinder and SpeakEasy indicate that (BC,BF)=(1,2) is the best. Thus, we could conclude that (BC,BF)=(1,2) is the best on Network Science coauthorship network.

Table A.9: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on PGP network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.5	0.5	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(1,2)	0.5	0.5	0.5	0.5	0.05	0.45	0.5	0.5	0.5	0.5	0.5	0.5	***0.5 (10)***
	(2,1)	0.05	0.5	0.2	0.5	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
	(2,2)	0.45	0.5	0.2	0.5	0.05	0.45	0.5	0.5	0.5	0.5	0.5	0.5	0.5 (8)
CFinder	(1,1)	4	3	3	3	3	3	3	13	3	3	3	3	3 (10)
	(1,2)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(2,1)	4	3	3	6	3	3	3	18	18	19	3	13	3 (6)
	(2,2)	3	3	3	6	3	3	3	18	18	14,15	3	3	3 (8)
SpeakEasy	(1,1)	0.05	0.25	0.85	0.9	0.05	0.7	0.05	0.75	0.85	0.85	0.85	0.85	0.85 (5)
	(1,2)	0.85	0.25	0.85	0.9	0.85	0.7	0.85	0.75	0.85	0.85	0.25	0.85	***0.85 (7)***
	(2,1)	0.05	0.8	0.85	0.9	0.05	0.7	0.05	0.75	0.75	0.85	0.85	0.75	{0.05,0.75,0.85} (3)
	(2,2)	0.85	0.8	0.85	0.9	0.85	0.7	0.85	0.6	0.75	0.75	0.2	0.75	0.85 (4)

A.2.9 PGP Network

This is the largest connected component of the network of users of the Pretty-Good-Privacy (PGP) algorithm for secure information interchange [82]. It has 10680 nodes and 24316 edges in total. Table A.9 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on PGP Network. It can be seen from the table that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.10 Political Blogs Network

This is a directed network of hyperlinks between weblogs on US politics, recorded in 2005 by Adamic and Glance [83]. There are 1224 nodes and 19022 edges. Table A.10 shows the best value of threshold r for SLPA and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on political blogs network. Results for CFinder are not provided because it has not finished running on this network for more than two

Table A.10: The best value of threshold r for SLPA and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on political blogs network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.5	0.5	0.4	0.5	0.35	0.3	0.4	0.45	0.5	0.5	0.5	0.5 (7)
	(1,2)	0.5	0.5	0.5	0.4	0.5	0.4	0.5	0.15	0.4	0.5	0.5	0.5	***0.5 (8)***
	(2,1)	0.5	0.5	0.05	0.4	0.5	0.4	0.3	0.4	0.45	0.5	0.5	0.5	0.5 (6)
	(2,2)	0.25	0.25	0.05	0.25	0.5	0.4	0.5	0.05	0.35	0.5	0.5	0.3	0.5 (4)
SpeakEasy	(1,1)	0.85	0.45	0.85	0.7	0.9	0.35	0.05	0.7	0.7	0.25	0.45	0.7	***0.7 (4)***
	(1,2)	0.85	0.45	0.85	0.7	0.9	0.35	0.9	0.7	0.7	0.25	0.45	0.7	***0.7 (4)***
	(2,1)	0.85	0.45	0.45	0.7	0.9	0.35	0.05	0.7	0.5	0.25	0.45	0.5	0.45 (3)
	(2,2)	0.8	0.8	0.45	0.7	0.9	0.35	0.9	0.7	0.5	0.4	0.6	0.5	{0.5,0.7,0.8,0.9} (2)

months processing many potential k-cliques resulting from dense connections. It can be seen from the table that both SLPA and SpeakEasy imply that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.11 Political Books Network

This is a network of books about US politics published around the time of the 2004 presidential election and sold by the online bookseller Amazon.com [84]. It has 105 nodes and 441 edges in total. Edges between books indicate frequent copurchasing of books by the same buyers. Table A.11 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on political books network. We could learn from the table that SLPA shows that the first version of belonging function is better than the second version, CFinder implies that the first version of belonging coefficient is better than the second one, and SpeakEasy indicates that (BC,BF)=(1,2) is the best among all four combinations. In summary, there are two out of three algorithms support that (BC,BF)=(1,2) is the best on political books network.

Table A.11: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on political books network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.5	0.5	0.2	0.4	0.05	0.4	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (7)***
	(1,2)	0.5	0.5	0.2	0.4	0.1	0.4	0.1	0.4	0.2	0.2	0.2	0.2	0.2 (5)
	(2,1)	0.5	0.5	0.1	0.4	0.05	0.4	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (7)***
	(2,2)	0.5	0.5	0.1	0.4	0.1	0.4	0.1	0.4	0.2	0.2	0.2	0.2	0.2 (4)
CFinder	(1,1)	4	3	3	3	3	3	3	6	3	3	3	3	***3 (10)***
	(1,2)	3	3	3	4	3	3	3	6	3	3	3	3	***3 (10)***
	(2,1)	4	4	3	4	3	3	3	6	6	3	3	3	3 (7)
	(2,2)	4	4	3	4	3	3	3	6	6	6	3	3	3 (6)
SpeakEasy	(1,1)	0.95	1	0.5	0.95	0.25	0.95	0.05	0.5	0.9	0.95	0.95	0.9	0.95 (5)
	(1,2)	0.95	0.55	0.5	0.95	0.95	0.95	0.9	0.5	0.9	0.95	0.95	0.9	***0.95 (6)***
	(2,1)	0.5	1	0.5	0.95	0.25	0.95	0.05	0.5	0.85	0.9	0.95	0.9	{0.5,0.95} (3)
	(2,2)	0.5	0.55	0.5	0.95	0.95	0.95	0.9	0.5	0.85	0.85	0.95	0.9	0.95 (4)

Table A.12: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Indian railway network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.15	0.5	0.5	0.5	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(1,2)	0.5	0.35	0.5	0.5	0.05	0.45	0.5	0.5	0.5	0.5	0.5	0.5	***0.5 (9)***
	(2,1)	0.05	0.5	0.15	0.5	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
	(2,2)	0.5	0.35	0.15	0.5	0.05	0.45	0.5	0.35	0.5	0.5	0.5	0.5	0.5 (7)
CFinder	(1,1)	6	3	4	5	3	3	3	3	3	3	3	3	***3 (9)***
	(1,2)	6	3	4	5	3	3	3	3	3	3	3	3	***3 (9)***
	(2,1)	6	6	4	5	3	3	3	10	10	10	3	4	3 (4)
	(2,2)	6	6	4	6	3	3	3	10	10	10	3	4	3 (4)
SpeakEasy	(1,1)	0.55	0.8	0.45	1	0.05	0.7	0.05	0.95	0.95	0.8	0.8	0.8	0.8 (4)
	(1,2)	0.45	0.7	0.45	1	0.8	0.7	0.8	0.95	0.55	0.8	0.8	0.8	***0.8 (5)***
	(2,1)	0.55	0.8	0.45	1	0.05	0.7	0.05	0.95	0.95	0.8	0.8	0.55	0.8 (3)
	(2,2)	0.45	0.8	0.45	1	0.8	0.7	0.8	0.95	0.55	0.55	0.8	0.55	0.8 (4)

A.2.12 Indian Railway Network

This network consists of nodes representing Indian railway stations, where two stations are connected by an edge if there exists at least one train-route such that both stations are scheduled stops on that route [85]. There are 297 nodes and 1213 edges. Table A.12 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function

Table A.13: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve quality metrics with four combinations of the two versions of belonging coefficient and two version of belonging function on Santa Fe Institute collaboration network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.4	0.5	0.4	0.5	0.05	0.5	0.05	0.5	0.4	0.4	0.4	0.4	0.4 (6)
	(1,2)	0.4	0.5	0.4	0.5	0.1	0.5	0.4	0.5	0.4	0.4	0.4	0.4	***0.4 (7)***
	(2,1)	0.4	0.5	0.1	0.5	0.05	0.5	0.05	0.5	0.4	0.4	0.4	0.4	0.4 (5)
	(2,2)	0.4	0.5	0.1	0.5	0.1	0.5	0.1	0.5	0.4	0.4	0.4	0.4	0.4 (5)
CFinder	(1,1)	3	3	3	3	3	3	3	5	4	3	3	3	3 (10)
	(1,2)	3	3	3	3	3	3	3	4	3	3	3	3	***3 (11)***
	(2,1)	3	3	3	4	3	3	3	5	5	5	3	5	3 (7)
	(2,2)	3	3	3	3	3	3	3	5	5	5	3	4	3 (8)
SpeakEasy	(1,1)	0.1	0.9	0.65	0.65	0.05	0.9	0.05	1	0.95	0.9	0.9	0.9	***0.9 (5)***
	(1,2)	0.55	0.9	0.65	0.35	0.35	0.9	0.9	1	0.95	0.9	0.9	0.35	***0.9 (5)***
	(2,1)	0.1	0.9	0.1	0.65	0.05	0.9	0.05	1	0.95	0.9	0.9	0.65	0.9 (4)
	(2,2)	0.1	0.9	0.1	0.65	0.35	0.9	0.9	0.65	0.65	1	0.9	0.65	{0.65,0.9} (4)

on Indian railway network. It can be seen from the table that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.2.13 Santa Fe Institute Collaboration Network

This is the largest connected component of the collaboration network of scientists at the Santa Fe Institute, an interdisciplinary research center in Santa Fe, New Mexico [78]. It has 118 nodes and 200 edges. Nodes in this network represent scientists in residence at the Santa Fe Institute during any part of calendar year 1999 or 2000 and their collaborators. An edge is drawn between a pair of scientists if they coauthored one or more articles during the same time period. The network includes all journal and book publications by the scientists involved, along with all papers that appeared in the institutes technical reports series. Table A.13 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Santa Fe Institute collaboration network. We could learn that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

Table A.14: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Collins_cyc. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.5	0.4	0.1	0.5	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (8)***
	(1,2)	0.5	0.45	0.5	0.5	0.2	0.5	0.2	0.5	0.4	0.2	0.2	0.4	0.5 (5)
	(2,1)	0.05	0.5	0.5	0.4	0.1	0.5	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (8)***
	(2,2)	0.5	0.5	0.5	0.5	0.2	0.5	0.2	0.5	0.4	0.2	0.2	0.4	0.5 (6)
CFinder	(1,1)	3	3	3	5	3	3	3	3	3	3	3	3	***3 (11)***
	(1,2)	3	3	3	5	3	3	3	3	3	3	3	3	***3 (11)***
	(2,1)	3	9	3	6	3	3	3	18	18	20	3	3	3 (7)
	(2,2)	3	3	3	6	3	3	3	17-20	17-20	17-20	3	3	3 (8)
SpeakEasy	(1,1)	0.05	0.9	0.9	1	0.9	0.9	0.05	0.8	0.9	0.9	0.9	0.9	0.9 (8)
	(1,2)	0.9	0.9	0.9	1	0.9	0.9	0.9	0.8	0.9	0.9	0.9	0.9	***0.9 (10)***
	(2,1)	0.05	0.55	0.9	0.95	0.05	0.9	0.05	0.8	0.95	0.9	0.9	0.95	0.9 (4)
	(2,2)	0.9	0.7	0.9	0.95	0.9	0.9	0.9	0.8	0.25	0.25	0.9	0.95	0.9 (6)

A.2.14 Protein-protein Interaction Networks

We consider eight protein-protein interaction networks in the experiments. **Collins_cyc**, **Collins_cyc_w**, **Collins_mips**, **Collins_sgd**, **Gavin_cyc**, **Gavin_cyc_w**, **Gavin_mips**, and **Gavin_sgd** are two kinds (referred as Collins [86] and Gavin [87] here) of popular high throughput protein-protein interaction networks derived from measurements obtained by affinity purification and mass spectrometry (AP-MS) techniques [74]. These two kinds of networks are further refined with three gold-standards for protein complexes, including the classic Munich Information Center for Protein Sequences (MIPS) [88] and the more recent Saccharomyces Genome Database (SGD) [89]. The complete MIPS dataset as well as partial information from SGD are incorporated into a third protein complex list known as CYC2008 [90]. Thus, we have **Collins_cyc**, **Collins_mips**, **Collins_sgd**, **Gavin_cyc**, **Gavin_mips**, and **Gavin_sgd**, respectively. **Collins_cyc_w** and **Gavin_cyc_w** are respectively the weighted versions of **Collins_cyc** and **Gavin_cyc**, in which the weight is proportional to the probability a given interaction pair truly exists.

Tables A.14-A.21 show the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging

Table A.15: The best value of threshold r for SLPA and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Collins_cyc_w. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.35	0.05	0.35	0.05	0.25	0.05	0.5	0.05	0.05	0.05	0.2	0.05 (7)
	(1,2)	0.05	0.35	0.05	0.35	0.05	0.25	0.4	0.05	0.05	0.05	0.05	0.35	0.05 (7)
	(2,1)	0.05	0.35	0.05	0.35	0.05	0.25	0.05	0.5	0.5	0.05	0.05	0.2	0.05 (6)
	(2,2)	0.05	0.5	0.05	0.35	0.05	0.25	0.05	0.05	0.05	0.05	0.05	0.2	***0.05 (8)***
SpeakEasy	(1,1)	0.1	0.55	0.55	0.9	0.1	0.65	0.1	0.95	0.55	0.55	0.55	0.55	0.55 (6)
	(1,2)	0.55	0.55	0.55	0.9	0.55	0.65	0.55	0.95	0.55	0.55	0.55	0.55	***0.55 (9)***
	(2,1)	0.1	0.4	0.55	0.9	0.1	0.65	0.1	0.95	0.8	0.55	0.55	0.45	{0.1,0.55} (3)
	(2,2)	0.55	0.4	0.55	0.9	0.55	0.65	0.55	0.8	0.8	0.8	0.7	0.45	0.55 (4)

Table A.16: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Collins_mips. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.15	0.45	0.45	0.45	0.05	0.5	0.05	0.5	0.45	0.15	0.05	0.45	0.45 (5)
	(1,2)	0.45	0.45	0.45	0.45	0.05	0.5	0.15	0.5	0.45	0.05	0.05	0.45	***0.45 (6)***
	(2,1)	0.15	0.45	0.05	0.45	0.05	0.5	0.05	0.5	0.5	0.45	0.05	0.45	{0.05,0.45} 4
	(2,2)	0.45	0.5	0.05	0.45	0.05	0.5	0.15	0.5	0.45	0.05	0.05	0.45	{0.05,0.45} (4)
CFinder	(1,1)	4	3	3	5	3	3	3	3	3	3	3	3	***3 (10)***
	(1,2)	4	3	3	5	3	3	3	3	3	3	3	3	***3 (10)***
	(2,1)	4	7	3	7	3	3	3	19	19	20	3	3	3 (6)
	(2,2)	4	7	3	7	3	3	3	16-20	16-20	16-20	3	3	3 (6)
SpeakEasy	(1,1)	0.05	0.4	0.4	1	0.4	0.4	0.35	0.6	0.7	0.4	0.4	0.4	0.4 (7)
	(1,2)	0.4	0.4	0.4	1	0.4	0.4	0.4	0.6	0.4	0.4	0.4	0.4	***0.4 (10)***
	(2,1)	0.05	0.4	0.25	1	0.4	0.4	0.35	0.6	0.75	0.4	0.4	0.75	0.4 (5)
	(2,2)	0.25	0.95	0.25	1	0.4	0.4	0.4	0.6	0.6	0.6	0.25	0.75	{0.25,0.4,0.6} (3)

function on the eight protein-protein interaction networks. Results for CFinder on **Collins_cyc_w** and **Gavin_cyc_w** are not provided because it has not finished running on **Collins_cyc_w** and **Gavin_cyc_w** for more than two months processing many potential k-cliques associated with intensity larger than the intensity threshold [120].

From Table A.14 we could see that on **Collins_cyc** SLPA implies that the first version of belonging function is better than the second version, CFinder indicates that the first version of belonging coefficient is better than the second one, and SpeakEasy demonstrates that (BC,BF)=(1,2) is the best among all four combinations. In summary, there are two out of three algorithms show that (BC,BF)=(1,2)

Table A.17: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Collins_sgd. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.35	0.45	0.05	0.5	0.05	0.5	0.5	0.35	0.35	0.45	0.5 (4)
	(1,2)	0.35	0.45	0.35	0.45	0.1	0.5	0.1	0.5	0.35	0.1	0.1	0.45	0.1 (4)
	(2,1)	0.05	0.5	0.1	0.45	0.05	0.5	0.05	0.5	0.5	0.35	0.35	0.45	0.5 (4)
	(2,2)	0.35	0.5	0.1	0.45	0.1	0.3	0.1	0.5	0.1	0.1	0.1	0.4	***0.1 (6)***
CFinder	(1,1)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(1,2)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(2,1)	3	3	3	6	3	3	3	16-20	16-20	16-20	3	3	3 (8)
	(2,2)	3	3	3	6	3	3	3	16-20	16-20	16-20	3	3	3 (8)
SpeakEasy	(1,1)	0.1	0.8	0.8	0.75	0.05	0.8	0.05	0.5	0.8	0.8	0.8	0.8	0.8 (7)
	(1,2)	0.8	0.8	0.8	0.75	0.8	0.8	0.8	0.35	0.8	0.8	0.8	0.8	***0.8 (10)***
	(2,1)	0.05	0.8	0.35	0.5	0.05	0.8	0.15	0.5	0.5	0.8	0.8	0.5	{0.5,0.8} (4)
	(2,2)	0.35	0.8	0.35	0.5	0.8	0.8	0.8	0.5	0.5	0.5	0.8	0.5	{0.5,0.8} (5)

Table A.18: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Gavin_cyc. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.45	0.5	0.45	0.05	0.5	0.05	0.5	0.5	0.45	0.45	0.45	0.45 (5)
	(1,2)	0.5	0.45	0.5	0.45	0.15	0.5	0.2	0.5	0.45	0.45	0.45	0.45	0.45 (6)
	(2,1)	0.05	0.45	0.2	0.45	0.05	0.5	0.05	0.5	0.5	0.45	0.45	0.45	0.45 (5)
	(2,2)	0.45	0.45	0.2	0.45	0.15	0.5	0.2	0.5	0.45	0.45	0.45	0.45	***0.45 (7)***
CFinder	(1,1)	3	3	3	3	3	3	3	4	3	3	3	3	3 (11)
	(1,2)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(2,1)	3	4	3	5	3	3	3	13	13	18-20	3	4	3 (6)
	(2,2)	3	4	3	5	3	3	3	11-20	11-20	11-20	3	4	3 (6)
SpeakEasy	(1,1)	0.25	0.95	0.7	0.85	0.05	1	0.05	0.7	0.7	0.7	0.7	0.7	0.7 (6)
	(1,2)	0.7	0.95	0.7	0.85	0.7	1	1	0.7	0.7	0.7	0.7	0.7	***0.7 (8)***
	(2,1)	0.1	0.95	0.7	0.85	0.05	1	0.05	1	0.7	0.7	0.7	0.7	0.7 (5)
	(2,2)	0.7	0.95	0.7	0.85	0.7	1	1	0.7	0.7	0.85	0.7	0.85	0.7 (6)

is the best on **Collins_cyc**. Also, SLPA in Table A.15 implies that (BC,BF)=(2,2) is the best, while SpeakEasy shows that (BC,BF)=(1,2) is the best on **Collins_cyc_w**. It can be seen from Table A.16 that all three algorithms support the conclusion that (BC,BF)=(1,2) is the best among the four combinations on **Collins_mips**. In addition on **Collins_sgd** (Table A.17), SLPA shows that (BC,BF)=(2,2) is the best, CFinder implies that the first version of belonging coefficient is better than the second version, and SpeakEasy indicates that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function. Thus, two

Table A.19: The best value of r for SLPA and the best value of tr for SpeakEasy determined by the twelve community quality metrics with four combinations of belonging coefficient and belonging function on Gavin_cyc_w. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.4	0.35	0.3	0.05	0.35	0.05	0.5	0.45	0.35	0.35	0.35	0.35 (5)
	(1,2)	0.35	0.4	0.35	0.3	0.05	0.35	0.3	0.45	0.3	0.3	0.3	0.3	***0.3 (6)***
	(2,1)	0.05	0.45	0.05	0.3	0.05	0.35	0.05	0.5	0.5	0.35	0.35	0.45	0.05 (4)
	(2,2)	0.3	0.45	0.05	0.3	0.05	0.35	0.3	0.45	0.5	0.5	0.3	0.3	0.3 (5)
SpeakEasy	(1,1)	0.05	0.4	0.45	0.65	0.05	0.65	0.1	0.95	0.7	0.7	0.7	0.7	0.7 (4)
	(1,2)	0.45	0.4	0.45	0.65	0.7	0.65	0.7	0.95	0.45	0.7	0.7	0.7	***0.7 (5)***
	(2,1)	0.05	0.7	0.45	0.5	0.05	0.65	0.1	0.95	0.95	0.5	0.7	1	{0.05,0.5,0.7,0.95} (2)
	(2,2)	0.45	0.7	0.45	0.65	0.7	0.65	0.7	0.95	0.95	0.9	0.4	1	0.7 (3)

out of three algorithms conclude that (BC,BF)=(1,2) is the best among the four combinations on **Collins_sgd**.

It can be observed from Table A.18 that on **Ganvin_cyc** SLPA shows that (BC,BF)=(2,2) is the best, while CFinder and SpeakEasy indicate that (BC,BF)=(1,2) is the best. Hence, two out of three algorithms support that (BC,BF)=(1,2) is the best among the four combinations on **Ganvin_cyc**. From Tables A.19-A.21, we could learn that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations on **Ganvin_cyc_w**, **Ganvin_mips**, and **Ganvin_sgd**.

From the analysis above, we could conclude that (BC,BF)=(1,2) is the best among the four possible combinations of the two versions of belonging coefficient and two versions of belonging function on protein-protein interaction networks.

A.2.15 Amazon Product Network

This is a product co-purchased network of the Amazon website [33]. If a product p_i is frequently co-purchased with product p_j, the graph contains an undirected edge from p_i to p_j. There are 319948 nodes and 880215 edges. Table A.22 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Amazon product network. We can see that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

Table A.20: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Gavin_mips. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.5	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
	(1,2)	0.5	0.5	0.5	0.5	0.1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	***0.5 (11)***
	(2,1)	0.05	0.5	0.25	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(2,2)	0.5	0.45	0.25	0.5	0.1	0.4	0.3	0.5	0.35	0.5	0.5	0.5	0.5 (6)
CFinder	(1,1)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(1,2)	3	3	3	3	3	3	3	3	3	3	3	3	***3 (12)***
	(2,1)	3	3	3	4	3	3	3	12	12	18-20	3	4	3 (7)
	(2,2)	3	4	3	4	3	3	3	11-20	11-20	11-20	3	4	3 (6)
SpeakEasy	(1,1)	0.05	0.4	0.9	0.7	0.05	0.95	0.1	0.9	0.9	0.9	0.9	0.9	0.9 (6)
	(1,2)	0.9	0.4	0.9	0.7	0.9	0.95	0.9	0.4	0.9	0.9	0.9	0.9	***0.9 (8)***
	(2,1)	0.05	0.95	0.9	0.7	0.05	0.95	0.1	0.9	0.9	0.9	0.9	0.4	0.9 (5)
	(2,2)	0.9	0.95	0.9	0.7	0.9	0.35	0.9	0.4	0.4	0.4	0.9	0.4	0.9 (5)

Table A.21: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on Gavin_sgd. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q^{L}_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.5	0.5	0.1	0.35	0.1	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(1,2)	0.5	0.5	0.5	0.5	0.1	0.35	0.5	0.35	0.5	0.5	0.5	0.5	***0.5 (9)***
	(2,1)	0.05	0.5	0.15	0.5	0.1	0.35	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
	(2,2)	0.5	0.45	0.15	0.5	0.1	0.35	0.5	0.35	0.5	0.5	0.5	0.5	0.5 (7)
CFinder	(1,1)	3	3	3	4	3	3	3	3	3	3	3	3	***3 (11)***
	(1,2)	3	3	3	4	3	3	3	3	3	3	3	3	***3 (11)***
	(2,1)	3	4	3	5	3	3	3	13	13	13	3	4	3 (6)
	(2,2)	3	4	3	5	3	3	3	11-17	11-17	11-17	3	4	3 (6)
SpeakEasy	(1,1)	0.05	0.8	0.45	0.5	0.8	0.8	0.15	0.7	0.45	0.8	0.8	0.45	0.8 (5)
	(1,2)	0.45	0.8	0.45	0.5	0.8	0.8	0.8	0.7	0.45	0.8	0.8	0.45	***0.8 (6)***
	(2,1)	0.05	0.8	0.8	0.5	0.8	0.8	0.15	0.7	0.45	0.8	0.8	0.45	***0.8 (6)***
	(2,2)	0.45	0.8	0.8	0.5	0.8	0.8	0.8	0.7	0.45	0.45	0.8	0.45	***0.8 (6)***

A.2.16 DBLP Collaboration Network

The DBLP computer science bibliography provides a comprehensive list of research papers in computer science. In this DBLP co-authorship network, two authors are connected if they publish at least one paper together [33]. It has 260998 nodes and 950059 edges in total. Table A.23 shows the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version

Table A.22: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve quality metrics with four combinations of the two versions of belonging coefficient and two version of belonging function on Amazon product network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q_{ov}^{L}	Q_{ds}^{ov}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.5	0.5	0.1	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
	(1,2)	0.5	0.5	0.5	0.5	0.5	0.4	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (10)***
	(2,1)	0.05	0.5	0.3	0.45	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (6)
	(2,2)	0.5	0.5	0.3	0.5	0.5	0.4	0.5	0.4	0.5	0.5	0.5	0.5	0.5 (9)
CFinder	(1,1)	3	3	3	3	3	3	3	7	3	3	3	3	***3 (11)***
	(1,2)	3	3	3	3	3	3	3	5	3	3	3	3	***3 (11)***
	(2,1)	3	4	3	4	3	3	3	7	7	7	3	5	3 (6)
	(2,2)	3	4	3	4	3	3	3	7	7	7	3	4	3 (6)
SpeakEasy	(1,1)	0.1	1	1	1	0.1	1	0.1	1	1	1	1	1	1 (9)
	(1,2)	1	1	1	1	1	1	1	1	1	0.95	1	1	***1 (11)***
	(2,1)	0.1	1	0.95	1	0.1	1	0.1	1	1	1	1	1	1 (8)
	(2,2)	1	1	0.95	1	1	1	1	1	1	1	1	1	***1 (11)***

Table A.23: The best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve quality metrics with four combinations of the two versions of belonging coefficient and two version of belonging function on DBLP collaboration network. The best value for subcolumn of the last column is marked by red italic font.

Algorithm	(BC,BF)	Q_{ov}	NQ_{ov}	Q_{ov}^{L}	Q_{ds}^{ov}	IE	ID	CNT	BE	EXP	CND	F	D	
SLPA	(1,1)	0.05	0.5	0.5	0.45	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
	(1,2)	0.5	0.5	0.5	0.5	0.5	0.45	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (10)***
	(2,1)	0.05	0.5	0.3	0.45	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (6)
	(2,2)	0.5	0.5	0.3	0.5	0.15	0.45	0.5	0.45	0.5	0.5	0.5	0.5	0.5 (8)
CFinder	(1,1)	4	3	3	3	3	3	3	18	7	3	3	3	3 (9)
	(1,2)	4	3	3	3	3	3	3	9	3	3	3	3	***3 (10)***
	(2,1)	4	5	3	6	3	3	3	20	19	20	3	16	3 (5)
	(2,2)	4	5	3	5	3	3	3	20	19	20	3	9	3 (5)
SpeakEasy	(1,1)	0.1	1	1	1	0.1	1	0.1	1	1	1	1	1	1 (9)
	(1,2)	1	0.95	1	1	1	1	0.95	1	1	1	1	1	***1 (10)***
	(2,1)	0.1	1	0.95	1	0.1	1	0.1	1	1	1	1	1	1 (8)
	(2,2)	1	1	0.95	1	1	1	0.95	1	1	1	1	1	***1 (10)***

of belonging function on DBLP collaboration network. We can see that all three algorithms show that (BC,BF)=(1,2) is the best among the four combinations of belonging coefficient and belonging function.

A.3 LFR Benchmark Networks

LFR (named after the initials of names of authors) benchmark networks [92] have become a standard in the evaluation of the performance of community detection

algorithms. The LFR benchmark network that we used here has 1000 nodes with average degree 15 and maximum degree 50. The exponent γ for the degree sequence varies from 2 to 3. The exponent β for the community size distribution ranges from 1 to 2. Then, four pairs of the exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1), \text{and } (3, 2)$ are chosen in order to explore the widest spectrum of graph structures. The mixing parameter μ is varied from 0.05 to 0.95. It means that each node shares a fraction $(1-\mu)$ of its edges with the other nodes in its community and shares a fraction μ of its edges with the nodes outside its community. Thus, low mixing parameters indicate strong community structure. The degree of overlap is determined by two parameters. O_n is the number of overlapping nodes, and O_m is the number of communities to which each overlapping node belongs. O_n here is set to 10% of the total number of nodes. Instead of fixing O_m, we allow it to vary from 1 to 8 indicating the overlapping diversity of overlapping nodes. By increasing the value of O_m, we create harder detection tasks. Also, we generate 10 network instances for each configuration of these parameters. Hence, each metric value for a certain configuration of LFR represents the average metric values of all 10 instances. Since the experimental results are similar for all four pairs of exponents $(\gamma, \beta) = (2, 1), (2, 2), (3, 1), \text{and } (3, 2)$, for the sake of brevity, we only present the results for $(\gamma, \beta) = (2, 1)$ here. In addition, there results are similar for different values of μ, so here we only show the results for $\mu = 0.3, 0.35, \text{and } 0.4$. We choose $\mu = 0.3, 0.35, \text{and } 0.4$ to better illustrate the results since with $\mu = 0.3, 0.35, \text{and } 0.4$ the community structure generated by LFR are around the boundary of well-separated communities and well-connected communities. For each node, $\mu = 0.5$ means that the number of its edges with other nodes in its communities is equal to the number of its edges with nodes outside its community, which makes the community structure difficult to discover.

Tables A.24-A.26 respectively show the best value of threshold r for SLPA, the best value of parameter k for CFinder, and the best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. Table A.24 implies that (BC,BF)=(1,2) is the best among the four possible combina-

Table A.24: The best value of threshold r for SLPA determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. The best value for subcolumn of the last column is marked by red italic font.

μ	O_m	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
0.3	1	(1,1)	0.3	0.5	0.5	0.5	0.05	0.45	0.05	0.45	0.5	0.5	0.5	0.5	0.5 (7)
		(1,2)	0.5	0.5	0.5	0.5	0.05	0.45	0.2	0.45	0.5	0.5	0.5	0.5	***0.5 (8)***
		(2,1)	0.3	0.5	0.2	0.5	0.05	0.5	0.05	0.45	0.5	0.5	0.5	0.5	0.5 (7)
		(2,2)	0.5	0.5	0.2	0.5	0.05	0.5	0.2	0.45	0.5	0.2	0.2	0.5	0.5 (6)
	2	(1,1)	0.05	0.5	0.5	0.25	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (8)***
		(1,2)	0.5	0.5	0.5	0.5	0.3	0.5	0.3	0.45	0.4	0.3	0.3	0.4	0.5 (5)
		(2,1)	0.05	0.5	0.3	0.25	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
		(2,2)	0.4	0.5	0.3	0.4	0.15	0.5	0.3	0.45	0.35	0.3	0.3	0.35	0.3 (4)
	4	(1,1)	0.05	0.5	0.5	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
		(1,2)	0.5	0.5	0.5	0.5	0.15	0.5	0.5	0.5	0.4	0.5	0.5	0.5	***0.5 (10)***
		(2,1)	0.05	0.5	0.3	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(2,2)	0.45	0.5	0.3	0.5	0.15	0.5	0.35	0.5	0.4	0.4	0.5	0.4	0.5 (5)
	6	(1,1)	0.05	0.5	0.5	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
		(1,2)	0.5	0.5	0.5	0.5	0.1	0.5	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (10)***
		(2,1)	0.05	0.5	0.3	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(2,2)	0.4	0.5	0.3	0.45	0.1	0.5	0.2	0.45	0.5	0.5	0.5	0.5	0.5 (6)
	8	(1,1)	0.1	0.5	0.5	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
		(1,2)	0.5	0.5	0.5	0.5	0.1	0.5	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (10)***
		(2,1)	0.1	0.5	0.2	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(2,2)	0.5	0.5	0.2	0.5	0.1	0.5	0.5	0.45	0.5	0.5	0.5	0.5	0.5 (9)
0.35	1	(1,1)	0.25	0.5	0.25	0.4	0.05	0.5	0.15	0.5	0.5	0.25	0.25	0.25	0.25 (5)
		(1,2)	0.25	0.5	0.25	0.4	0.1	0.45	0.15	0.45	0.25	0.25	0.25	0.25	***0.25 (6)***
		(2,1)	0.2	0.5	0.15	0.4	0.05	0.45	0.05	0.5	0.25	0.25	0.25	0.25	0.25 (4)
		(2,2)	0.25	0.5	0.15	0.4	0.05	0.45	0.15	0.45	0.25	0.25	0.25	0.25	0.25 (5)
	2	(1,1)	0.05	0.5	0.45	0.3	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (7)***
		(1,2)	0.45	0.5	0.45	0.45	0.15	0.45	0.3	0.45	0.45	0.3	0.3	0.45	***0.45 (7)***
		(2,1)	0.05	0.5	0.3	0.25	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	***0.5 (7)***
		(2,2)	0.45	0.5	0.3	0.45	0.15	0.45	0.15	0.45	0.45	0.3	0.3	0.45	0.45 (6)
	4	(1,1)	0.05	0.5	0.5	0.5	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(1,2)	0.5	0.5	0.5	0.5	0.15	0.45	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (9)***
		(2,1)	0.05	0.5	0.3	0.5	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
		(2,2)	0.5	0.5	0.3	0.45	0.1	0.45	0.5	0.45	0.45	0.5	0.5	0.5	0.5 (6)
	6	(1,1)	0.05	0.5	0.5	0.45	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(1,2)	0.5	0.5	0.5	0.5	0.1	0.45	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (9)***
		(2,1)	0.05	0.5	0.25	0.45	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
		(2,2)	0.5	0.5	0.25	0.5	0.1	0.45	0.5	0.45	0.5	0.5	0.5	0.5	0.5 (8)
	8	(1,1)	0.35	0.5	0.5	0.45	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
		(1,2)	0.5	0.5	0.5	0.5	0.05	0.45	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (9)***
		(2,1)	0.4	0.5	0.2	0.45	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (6)
		(2,2)	0.5	0.5	0.2	0.5	0.05	0.45	0.5	0.45	0.5	0.5	0.5	0.5	0.5 (8)
0.4	1	(1,1)	0.25	0.5	0.5	0.45	0.05	0.45	0.1	0.5	0.5	0.25	0.5	0.5	***0.5 (6)***
		(1,2)	0.5	0.45	0.5	0.45	0.1	0.4	0.25	0.4	0.25	0.25	0.25	0.25	0.25 (5)
		(2,1)	0.2	0.5	0.2	0.45	0.05	0.45	0.05	0.5	0.5	0.25	0.25	0.5	0.5 (4)
		(2,2)	0.45	0.45	0.2	0.45	0.1	0.4	0.2	0.4	0.25	0.2	0.25	0.25	{0.2,0.25,0.45} (3)
	2	(1,1)	0.05	0.5	0.5	0.3	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(1,2)	0.5	0.5	0.5	0.5	0.15	0.5	0.5	0.4	0.45	0.5	0.5	0.5	***0.5 (9)***
		(2,1)	0.05	0.5	0.3	0.3	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
		(2,2)	0.45	0.5	0.3	0.5	0.15	0.4	0.5	0.35	0.45	0.5	0.5	0.45	0.5 (5)
	4	(1,1)	0.1	0.5	0.5	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
		(1,2)	0.5	0.5	0.5	0.5	0.2	0.5	0.5	0.4	0.5	0.5	0.5	0.5	***0.5 (10)***
		(2,1)	0.1	0.5	0.3	0.5	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(2,2)	0.5	0.5	0.3	0.5	0.15	0.5	0.5	0.4	0.45	0.5	0.5	0.5	0.5 (8)
	6	(1,1)	0.5	0.5	0.5	0.45	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
		(1,2)	0.5	0.5	0.5	0.5	0.05	0.5	0.5	0.45	0.5	0.5	0.5	0.5	***0.5 (10)***
		(2,1)	0.5	0.5	0.5	0.45	0.05	0.5	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (9)
		(2,2)	0.5	0.5	0.5	0.5	0.05	0.45	0.5	0.4	0.5	0.5	0.5	0.5	0.5 (9)
	8	(1,1)	0.5	0.5	0.5	0.4	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (8)
		(1,2)	0.5	0.5	0.5	0.5	0.1	0.45	0.5	0.4	0.5	0.5	0.5	0.5	***0.5 (9)***
		(2,1)	0.5	0.5	0.25	0.4	0.05	0.45	0.05	0.5	0.5	0.5	0.5	0.5	0.5 (7)
		(2,2)	0.5	0.5	0.25	0.5	0.05	0.4	0.5	0.4	0.5	0.5	0.5	0.5	0.5 (8)

Table A.25: The best value of parameter k for CFinder determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. The best value for subcolumn of the last column is marked by red italic font.

μ	O_m	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
0.3	1	(1,1)	4	4	4	4	3	3	4	10	4	4	4	4	***4 (9)***
		(1,2)	4	4	4	4	3	3	4	10	4	4	4	4	***4 (9)***
		(2,1)	4	5	4	4	3	3	4	10	10	4	4	4	4 (7)
		(2,2)	4	5	4	4	3	3	4	10	10	10	4	4	4 (6)
	2	(1,1)	4	5	4	4	3	3	3	9	4	3	3	4	{3,4} (5)
		(1,2)	4	5	4	4	3	3	4	9	4	3	3	4	***4 (6)***
		(2,1)	4	5	4	4	3	3	3	9	9	3	3	4	3 (5)
		(2,2)	4	6	4	4	3	3	4	9	9	9	3	4	4 (5)
	4	(1,1)	4	5	4	4	3	3	3	12	4	3	3	3	***3 (6)***
		(1,2)	4	5	4	4	3	3	3	12	4	3	3	3	***3 (6)***
		(2,1)	4	6	4	4	3	3	3	12	12	12	3	9	3 (4)
		(2,2)	4	6	4	4	3	3	3	12	12	12	3	7	3 (4)
	6	(1,1)	4	5	4	4	3	3	4	11	5	3	3	3	3 (5)
		(1,2)	4	5	4	4	3	3	4	11	4	4	4	3	***4 (7)***
		(2,1)	4	7	4	5	3	3	4	11	11	11	3	7	{3,4,11} (3)
		(2,2)	4	7	4	5	3	3	4	11	11	11	4	7	4 (4)
	8	(1,1)	4	5	4	4	3	3	4	12	6	3	3	3	3 (5)
		(1,2)	4	5	4	4	3	3	4	12	4	4	4	3	***4 (7)***
		(2,1)	4	6	4	5	3	3	4	12	12	12	3	7	{3,4,12} (3)
		(2,2)	4	6	4	5	3	3	4	12	12	12	4	6	4 (4)
0.35	1	(1,1)	4	4	4	4	3	3	3	10	4	4	4	4	***4 (8)***
		(1,2)	4	4	4	4	3	3	4	10	4	3	4	4	***4 (8)***
		(2,1)	4	4	4	4	3	3	3	10	10	10	4	4	4 (6)
		(2,2)	4	7	4	4	3	3	4	10	10	10	3	4	4 (5)
	2	(1,1)	4	4	4	4	3	3	3	8	4	3	3	3	***3 (6)***
		(1,2)	4	4	4	4	3	3	3	8	4	3	3	3	***3 (6)***
		(2,1)	4	5	4	4	3	3	3	8	8	8	3	8	{3,8} (4)
		(2,2)	4	7	4	4	3	3	4	8	8	8	3	7	4 (4)
	4	(1,1)	4	5	4	4	3	3	3	10	4	3	3	3	***3 (6)***
		(1,2)	4	4	4	4	3	3	4	10	4	3	3	3	***4 (6)***
		(2,1)	4	6	4	4	3	3	3	10	10	10	3	9	3 (4)
		(2,2)	4	8	4	4	3	3	4	10	10	10	3	7	4 (4)
	6	(1,1)	4	5	4	4	3	3	4	11	5	4	3	3	4 (5)
		(1,2)	4	5	4	4	3	3	4	11	4	4	4	3	***4 (7)***
		(2,1)	4	6	4	4	3	3	4	11	11	11	4	8	4 (5)
		(2,2)	4	7	4	4	3	3	4	11	11	11	4	7	4 (5)
	8	(1,1)	4	5	4	4	3	3	4	11	6	4	4	3	4 (6)
		(1,2)	4	5	4	4	3	3	4	11	4	4	4	3	***4 (7)***
		(2,1)	4	6	4	5	3	3	4	11	11	11	4	6	4 (4)
		(2,2)	4	6	4	4	3	3	4	11	11	11	4	6	4 (5)
0.4	1	(1,1)	4	4	4	4	3	3	3	8	4	3	3	4	***4 (6)***
		(1,2)	4	4	4	4	3	3	3	8	4	3	3	3	***3 (6)***
		(2,1)	4	4	4	4	3	3	3	8	8	8	3	8	{3,4,8} (4)
		(2,2)	4	7	4	4	3	3	4	8	8	8	3	7	4 (4)
	2	(1,1)	4	4	4	4	3	3	3	9	4	3	3	3	***3 (6)***
		(1,2)	4	4	4	4	3	3	3	9	4	3	3	3	***3 (6)***
		(2,1)	4	5	4	4	3	3	3	9	9	9	3	9	{3,9} (4)
		(2,2)	4	7	4	4	3	3	4	9	9	9	3	7	4 (4)
	4	(1,1)	4	4	4	4	3	3	3	9	5	3	3	3	***3 (6)***
		(1,2)	4	4	4	4	3	3	4	9	4	3	3	3	***4 (6)***
		(2,1)	4	5	4	4	3	3	3	9	9	9	3	9	{3,9} (4)
		(2,2)	4	7	4	4	3	3	4	9	9	9	3	7	4 (4)
	6	(1,1)	4	4	4	4	3	3	4	10	5	4	4	3	4 (7)
		(1,2)	4	4	4	4	3	3	4	10	4	4	4	3	***4 (8)***
		(2,1)	4	5	4	4	3	3	4	10	10	10	4	10	4 (5)
		(2,2)	4	7	4	4	3	3	4	10	10	10	4	7	4 (5)
	8	(1,1)	4	5	4	4	3	3	4	10	5	4	4	3	4 (6)
		(1,2)	4	5	4	4	3	3	4	10	4	4	4	3	***4 (7)***
		(2,1)	4	5	4	4	3	3	4	10	10	10	4	9	4 (5)
		(2,2)	4	6	4	4	3	3	4	10	10	10	4	6	4 (5)

Table A.26: The best value of threshold tr for SpeakEasy determined by the twelve community quality metrics with four possible combinations of the two versions of belonging coefficient and two version of belonging function on LFR benchmark networks with $(\alpha, \beta) = (1, 2)$ and $\mu = 0.3, 0.35, 0.4$. The best value for subcolumn of the last column is marked by red italic font.

μ	O_m	(BC,BF)	Q_{ov}	NQ_{ov}	Q^L_{ov}	Q^{ov}_{ds}	IE	ID	CNT	BE	EXP	CND	F	D	
0.3	1	(1,1)	0.75	0.2	0.75	0.75	0.75	0.8	0.75	0.8	0.75	0.75	0.75	0.75	***0.75 (9)***
		(1,2)	0.75	0.2	0.75	0.75	0.75	0.8	0.75	0.8	0.75	0.75	0.75	0.75	***0.75 (9)***
		(2,1)	0.75	0.2	0.75	0.75	0.75	0.8	0.75	0.8	0.75	0.75	0.75	0.75	***0.75 (9)***
		(2,2)	0.75	0.2	0.75	0.75	0.75	0.8	0.75	0.8	0.75	0.75	0.75	0.75	***0.75 (9)***
	2	(1,1)	0.05	0.85	0.8	0.6	0.8	0.4	0.8	0.9	0.8	0.8	0.8	0.8	0.8 (7)
		(1,2)	0.8	0.85	0.8	0.6	0.8	0.25	0.8	0.4	0.8	0.8	0.8	0.8	***0.8 (8)***
		(2,1)	0.05	0.85	0.8	0.6	0.8	0.4	0.8	0.9	1	0.8	0.8	0.8	0.8 (6)
		(2,2)	0.8	0.85	0.8	0.6	0.8	0.25	0.8	0.4	1	0.8	0.8	0.8	0.8 (7)
	4	(1,1)	0.05	1	0.7	0.95	0.05	1	0.05	1	1	0.8	0.8	0.95	***1 (4)***
		(1,2)	0.7	1	0.7	0.95	0.2	1	0.6	1	1	0.65	0.65	0.6	***1 (4)***
		(2,1)	0.05	1	0.7	0.95	0.05	1	0.05	1	1	0.8	0.8	0.95	***1 (4)***
		(2,2)	0.25	1	0.7	0.45	0.2	1	0.6	1	1	0.45	0.65	0.2	***1 (4)***
	6	(1,1)	0.05	0.85	0.5	1	0.05	0.65	0.05	0.7	0.95	0.95	0.95	0.95	***0.95 (4)***
		(1,2)	0.35	0.85	0.5	0.5	0.35	0.65	0.35	0.6	0.95	0.95	0.95	0.95	***0.95 (4)***
		(2,1)	0.05	0.85	0.35	1	0.05	0.65	0.05	0.7	0.7	0.95	0.95	1	0.05 (3)
		(2,2)	0.35	0.85	0.35	0.5	0.35	0.65	0.35	0.7	0.7	0.7	0.95	0.7	***{0.35,0.7} (4)***
	8	(1,1)	0.05	1	0.4	0.85	0.05	0.95	0.05	0.9	0.9	0.6	0.75	1	0.05 (3)
		(1,2)	0.4	1	0.4	0.85	0.4	0.95	0.4	0.35	0.6	0.6	0.6	0.6	***{0.4,0.6} (4)***
		(2,1)	0.05	1	0.4	0.85	0.05	0.95	0.05	0.9	0.85	0.6	0.75	0.85	{0.05,0.85} (3)
		(2,2)	0.35	1	0.4	0.85	0.4	0.95	0.4	0.35	0.85	0.7	0.6	0.85	{0.4,0.85} (3)
0.35	1	(1,1)	0.8,0.9	0.75	0.8	0.8,0.9	0.8,0.9	0.45	0.8,0.9	0.45	0.8,0.9	0.8,0.9	0.8,0.9	0.8,0.9	***0.8 (9)***
		(1,2)	0.8,0.9	0.75	0.8	0.8,0.9	0.8,0.9	0.45	0.8,0.9	0.45	0.8,0.9	0.8,0.9	0.8,0.9	0.8,0.9	***0.8 (9)***
		(2,1)	0.8,0.9	0.75	0.8	0.8,0.9	0.8,0.9	0.45	0.8,0.9	0.45	0.8,0.9	0.8,0.9	0.8,0.9	0.8,0.9	***0.8 (9)***
		(2,2)	0.8,0.9	0.75	0.8	0.8,0.9	0.8,0.9	0.45	0.8,0.9	0.45	0.8,0.9	0.8,0.9	0.8,0.9	0.8,0.9	***0.8 (9)***
	2	(1,1)	0.05	0.85	0.95	1	0.05	0.9	0.05	0.9	0.95	0.95	0.95	0.95	0.95 (5)
		(1,2)	0.95	0.85	0.95	1	0.35	0.9	0.95	0.5	0.75	0.95	0.95	0.95	***0.95 (6)***
		(2,1)	0.05	0.85	0.35	1	0.05	0.9	0.05	0.9	0.95	0.95	0.95	0.95	0.95 (4)
		(2,2)	0.35	0.85	0.35	1	0.35	0.9	0.95	0.5	0.75	0.75	0.95	0.95	{0.35,0.95} (3)
	4	(1,1)	0.05	0.95	0.9	1	0.05	0.95	0.05	0.95	0.9	0.75	0.75	1	{0.05,0.95} (3)
		(1,2)	0.9	0.95	0.9	1	0.65	0.95	0.75	0.95	0.9	0.75	0.75	0.75	***0.75 (4)***
		(2,1)	0.05	0.95	0.65	1	0.05	0.95	0.05	0.95	0.9	0.75	0.75	0.9	{0.05,0.95} (3)
		(2,2)	0.9	0.95	0.65	1	0.65	0.95	0.75	0.95	0.6	0.6	0.75	0.9	0.95 (3)
	6	(1,1)	0.05	1	0.8	0.85	0.05	1	0.05	1	0.85	0.85	0.85	0.85	***0.85 (5)***
		(1,2)	0.8	0.95	0.8	0.45	0.3	1	0.3	0.1	0.85	0.85	0.85	0.85	0.85 (4)
		(2,1)	0.05	1	0.45	0.85	0.05	1	0.05	1	0.9	0.85	0.85	0.85	0.85 (4)
		(2,2)	0.45	0.95	0.45	0.45	0.3	1	0.3	0.35	0.9	0.9	0.85	0.85	0.45 (3)
	8	(1,1)	0.05	1	0.9	0.8	0.05	0.85	0.05	1	0.95	0.85	0.85	0.9	{0.05,0.85} (3)
		(1,2)	0.9	1	0.9	0.8	0.9	0.85	0.85	0.3	0.9	0.85	0.85	0.9	***0.9 (5)***
		(2,1)	0.05	1	0.9	0.8	0.05	0.85	0.05	1	1	0.85	0.85	0.95	{0.05,0.85,1} (3)
		(2,2)	0.9	1	0.9	0.8	0.9	0.85	0.85	0.3	0.2	1	0.85	0.95	{0.85,0.9} (3)
0.4	1	(1,1)	0.15	0.75	0.15	0.15	0.15	0.75	0.15	0.75	0.15	0.15	0.15	0.15	***0.15 (9)***
		(1,2)	0.15	0.2	0.15	0.15	0.15	0.75	0.15	0.75	0.15	0.15	0.15	0.15	***0.15 (9)***
		(2,1)	0.15	0.75	0.15	0.15	0.15	0.75	0.15	0.75	0.15	0.15	0.15	0.15	***0.15 (9)***
		(2,2)	0.15	0.2	0.15	0.15	0.15	0.75	0.15	0.75	0.15	0.15	0.15	0.15	***0.15 (9)***
	2	(1,1)	0.05	0.85	0.95	0.8	0.1	0.85	0.05	0.85	0.95	0.8	0.8	0.95	{0.8,0.85,0.95} (3)
		(1,2)	0.95	0.85	0.95	0.75	0.8	0.85	0.75	0.85	0.95	0.8	0.8	0.95	0.95 (4)
		(2,1)	0.05	0.85	0.95	0.75	0.05	0.85	0.05	0.85	0.95	0.8	0.8	0.95	{0.05,0.85,0.95} (3)
		(2,2)	0.95	0.85	0.95	0.75	0.8	0.85	0.75	0.6	0.95	0.95	0.8	0.95	***0.95 (5)***
	4	(1,1)	0.05	1	0.95	0.9	0.15	1	0.05	0.9	0.95	0.95	0.95	0.95	0.95 (5)
		(1,2)	0.95	1	0.95	0.9	0.5	1	0.95	0.9	0.75	0.95	0.95	0.95	***0.95 (6)***
		(2,1)	0.05	1	0.95	0.9	0.05	1	0.05	0.9	0.9	0.95	0.95	0.95	0.95 (4)
		(2,2)	0.95	0.9	0.95	0.9	0.5	1	0.95	0.9	0.7	0.7	0.95	0.7	0.95 (4)
	6	(1,1)	0.1	0.95	0.75	0.75	0.15	0.95	0.1	0.95	0.95	0.9	0.9	0.8	0.95 (4)
		(1,2)	0.75	0.95	0.75	0.75	0.15	0.95	0.65	0.45	0.8	0.65	0.65	0.8	{0.65,0.75} (3)
		(2,1)	0.05	0.95	0.75	0.75	0.05	0.95	0.1	0.95	0.75	0.9	0.9	0.75	0.75 (4)
		(2,2)	0.75	0.95	0.75	0.75	0.15	0.95	0.65	0.45	0.75	0.75	0.65	0.75	***0.75 (6)***
	8	(1,1)	0.05	1	0.9	0.8	0.05	1	0.05	1	0.9	0.95	0.95	0.9	{0.05,0.9,1} (3)
		(1,2)	0.9	1	0.9	0.8	0.4	1	0.4	0.45	0.9	0.4	0.4	0.4	***0.4 (5)***
		(2,1)	0.05	1	0.9	0.8	0.05	1	0.05	1	0.8	0.95	0.95	0.95	{0.05,0.95,1} (3)
		(2,2)	0.9	1	0.9	0.8	0.4	1	0.4	0.45	0.5	0.5	0.4	0.5	{0.4,0.5} (3)

tions on all configurations of LFR benchmark networks except $\mu = 0.3, O_m = 2$ and $\mu = 0.4, O_m = 1$ when using SLPA. Table A.25 demonstrates that (BC,BF)=(1,2) is the best on all configurations of LFR benchmark networks when using CFinder. Table A.26 indicates that (BC,BF)=(1,2) is the best among the four combinations on all configurations of LFR benchmark networks except $\mu = 0.35, O_m = 6$ and $\mu = 0.4, O_m = 2, 4$ when using SpeakEasy. Consequently, we could conclude that the overlapping community quality metrics with the first version of belonging coefficient and the second version of the belonging function are the best among the four possible combinations on LFR benchmark networks.