# IDLE: A Novel Approach to Improving Overlapping Community Detection in Complex Networks

Rathna Senthil

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

February 16, 2016
Blacksburg, Virginia

Keywords: Overlapping Community Detection, Complex Networks, Local Expansion

# IDLE: A Novel Approach to Improving Overlapping Community Detection in Complex Networks

Rathna Senthil

(ABSTRACT)

Complex systems in areas such as biology, physics, social science, and technology are extensively modeled as networks due to the rich set of tools available for their study and analysis. In such networks, groups of nodes that correspond to functional units or those that share some common attributes result in densely connected structures called communities. Community formation is an inherent process, and it is not easy to detect these structures because of the complex ways in which components of these systems interact.

Detecting communities in complex networks is important because it helps us to understand their internal dynamics better, thereby leading to significant insights into the underlying systems. Overlapping communities are formed when nodes in the network simultaneously belong to more than one community, and it has been shown that most real networks naturally contain such an overlapping community structure. In this thesis, I introduce a new approach to overlapping community detection called IDLE that incorporates ideas from another interesting problem: the identification of influential spreaders. Influential spreaders are nodes that play an important role in the propagation of information or diseases in networks. Research suggests that the main core identified by $k$-core decomposition techniques are the most influential spreaders. In my approach, I use these $k$-cores as candidate seeds for local community detection. Following a well-defined seed selection process, IDLE builds and prunes their corresponding local communities. It then augments the resulting local communities and puts them together to obtain the global overlapping community structure of the network.

My approach improves on the current local community detection techniques, because they use either random nodes or maximal $k$-cliques as seeds, and they do not focus explicitly on detecting overlapping nodes in the network. Hence their results can be significantly improved in building ground-truth overlapping communities. The results of my experiments on real and synthetic networks indicate that IDLE results in enhanced overlapping community detection and thereby a better identification of overlapping nodes that could be important or influential components in the underlying system.

# IDLE: A Novel Approach to Improving Overlapping Community Detection in Complex Networks

Rathna Senthil

(GENERAL AUDIENCE ABSTRACT)

Complex systems in areas such as biology, physics, social science, and technology are commonly represented as networks due to the rich set of tools available for their study and analysis. Some examples of such complex networks are protein interaction networks, collaboration networks, social networks, network of telephone interactions, and the World Wide Web. The nodes in these networks represent entities in the system and the edges between them represent interactions or associations. In such networks, groups of nodes that share dense internal connections(edges) among themselves and sparse external connections with the rest of the network form communities. Community detection is the process of identifying such node clusters which could correspond to a functional unit or could be the result of assortative mixing of entities in the underlying system.

Overlapping community detection is the process of identifying communities where nodes in the network can belong to more than one community. In this thesis, I introduce a new approach to overlapping community detection called IDLE that incorporates ideas from another interesting problem: identification of the most influential nodes in a network. These are the nodes that play an important role in the propagation of information or diseases in networks. In my approach, I use these influential nodes as seeds for local community detection. Starting from multiple seed nodes, local communities are discovered until the entire network is covered. IDLE also distinguishes nodes as core nodes and overlapping nodes as it progresses, and this is found to result in a better detection of overlapping communities.

Identifying overlapping communities and, in turn, overlapping nodes in networks would uncover a wealth of information about the underlying complex systems. In spite of a number of challenges, many overlapping community detection techniques have been developed, motivated by the prospect of valuable insights one can get from this analysis. The results of my experiments on real and synthetic networks indicate that IDLE results in enhanced overlapping community detection and thereby a better identification of overlapping nodes that could be important or influential entities in the underlying system.

# Acknowledgments

I would like to thank my advisor Dr. Lenwood S. Heath for being the mentor I needed and steering me in the right direction all through my Master's at Virginia Tech. He has always motivated me to explore and investigate before deciding the path and this is one important lesson I have learned under his guidance. He has consistently supported my decisions in research and has helped me tremendously in improving my research abilities, including expressing it in writing.

I would like to thank Dr. B. Aditya Prakash and Dr. Sharath Raghvendra, from whom I have learned a lot, not only from our interactions regarding research, but also from the discussions and the brainstorming sessions I have had with them for my projects in related courses. Their valuable suggestions have played an important part in improving this thesis.

I would like to thank my friends and roommates for looking out for me and always being there when I needed them.

I express my sincere gratitude to my husband, parents, in-laws, and my wonderful family for constantly encouraging and supporting me to pursue my dreams and achieve them, no matter what.

# Contents

**1 Introduction** **1**

1.1 Motivation . . . 1

1.2 Problem . . . 2

1.3 Reader's Guide . . . 4

**2 Preliminaries** **5**

**3 Related Work** **8**

**4 Improved Detection Using Local Expansion** **11**

4.1 Seeds Selection . . . 11

4.2 Fitness Function . . . 15

4.3 Local Expansion . . . 17

4.4 Pruning . . . 19

4.5 Post-Processing . . . 22

4.6 Optimizations . . . 24

**5 Experiments and Results** **28**

5.1 Tests on Synthetic Networks . . . 28

5.1.1 LFR Benchmarks . . . 29

5.1.2 Normalized Mutual Information . . . 30

5.1.3 Omega Index . . . 31

5.1.4 Effects of Benchmark Parameters on Quality of Detection . . . 32

5.1.5 Detection of Overlapping Communities . . . 36

5.1.6 Detection of Overlapping Nodes . . . . . 42
5.1.7 Detection of Overlapping Memberships . . . . . 47
5.1.8 Scalability . . . . . 49
5.2 Tests on Real Networks . . . . . 50
5.2.1 Extended Modularity for Overlapping Community Detection . . . . . 51
5.2.2 Selecting an Ideal Extended Modularity . . . . . 53
5.2.3 Results on Real Networks . . . . . 58

**6 Conclusions and Future Work** **62**

**Bibliography** **64**

# List of Figures

1.1 Zachary's Karate Club Network showing the two communities in blue and red . . . . 3

2.1 Partition of a network. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

2.2 Cover obtained for a network. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

4.1 The $k$-core decomposition of Zachary's Karate Club Network . . . . . . . . . . . . . 12

4.2 The $k$-core decomposition of a large LFR Benchmark Network . . . . . . . . . . . . 13

4.3 Algorithm $K$-CORE DECOMPOSITION . . . . . . . . . . . . . . . . . . . . . . . . 14

4.4 Algorithm LOCAL-EXPANSION . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20

4.5 Algorithm PRUNE . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

4.6 Algorithm POST-PROCESSING . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

4.7 Algorithm IDLE . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

5.1 Effects of network size and mixing parameter on peformance . . . . . . . . . . . . . 33

5.2 Effects of community sizes and number of overlapping nodes on performance . . . . 35

5.3 Comparative study plots showing the performances of IDLE, GCE and SLPA on networks with $N$=5000, $s$=[10,50], $\mu$=0.3, $O_n$=10%. . . . . . . . . . . . . . . . . . . 36

5.4 Comparative study plots showing the performances of IDLE, GCE and SLPA on networks with $N$=5000, $b$=[20,100], $\mu$=0.3, $O_n$=10%. . . . . . . . . . . . . . . . . . . 37

5.5 Comparative study plots showing the performances of IDLE, GCE and SLPA on networks with $N$=5000, $s$=[10,50], $\mu$=0.3, $O_n$=50%. . . . . . . . . . . . . . . . . . . 38

5.6 Comparative study plots showing the performances of IDLE, GCE and SLPA on networks with $N$=5000, $b$=[20,100], $\mu$=0.3, $O_n$=50%. . . . . . . . . . . . . . . . . . . 39

5.7 Comparative study plots showing the performances of IDLE, GCE and SLPA on networks with $N$=5000, $b$=[20,100], $\mu$=0.1, $O_m$=4 for increasing $O_n$. . . . . . . . . . 41

5.8 Precision, Recall and F1 scores of IDLE, GCE and SLPA in detecting overlapping nodes in networks with $N$=5000, $s$=[10,50], $\mu$=0.3, $O_n$=10% . . . . . . . . . . . . . 43

5.9 Precision, Recall and F1 scores of IDLE, GCE and SLPA in detecting overlapping nodes in networks with $N$=5000, $b$=[20,100], $\mu$=0.3, $O_n$=10% . . . . . . . . . . . . . 44

5.10 Precision, Recall and F1 scores of IDLE, GCE and SLPA in detecting overlapping nodes in networks with $N$=5000, $s$=[10,50], $\mu$=0.3, $O_n$=50% . . . . . . . . . . . . . 45

5.11 Precision, Recall and F1 scores of IDLE, GCE and SLPA in detecting overlapping nodes in networks with $N$=5000, $b$=[20,100], $\mu$=0.3, $O_n$=50% . . . . . . . . . . . . . 46

5.12 Comparative study plots showing the performances of IDLE, GCE and SLPA in detecting overlapping memberships on networks, for increasing $O_m$. . . . . . . . . . . . 48

5.13 Time taken by IDLE on networks of increasing size . . . . . . . . . . . . . . . . . . . . 49

5.14 Time taken by IDLE on networks of increasing average degree for nodes . . . . . . . 50

5.15 Time taken by IDLE, GCE and SLPA on some real networks . . . . . . . . . . . . . 60

# List of Tables

4.1 Table showing the nodes in Zachary's Karate Club network grouped according to their k-core values and degrees . . . . . 13

4.2 Table showing the parameters defined for IDLE . . . . . 26

5.1 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 1, for ground-truth cover and Louvain partition 55

5.2 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 1, for covers from IDLE, GCE and SLPA . . . . 55

5.3 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 2, for ground-truth cover and Louvain partition 56

5.4 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 2, for covers from IDLE, GCE and SLPA . . . . 56

5.5 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 3, for ground-truth cover and Louvain partition 57

5.6 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 3, for covers from IDLE, GCE and SLPA . . . . 57

5.7 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 4, for ground-truth cover and Louvain partition 58

5.8 Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 4, for covers from IDLE, GCE and SLPA . . . . 58

5.9 Table showing the properties of real networks used in experiments . . . . . 59

5.10 Table showing the values of $Q_{ov}$ and $Q_{ds}^{ov}$ for real networks . . . . . 60

# Chapter 1

# Introduction

## 1.1 Motivation

Many systems in areas as diverse as physics, biology, social science, and technology, among others, are modeled as networks because of their relative ease of analysis. Some examples of such complex networks are collaboration networks [10, 23, 53], metabolic networks [32, 71], protein interaction networks [31, 69], network of telephone interactions [3, 39], and the World Wide Web [9, 15, 36]. The structure and properties of complex networks are studied extensively by Watts [72], Strogatz [67], Barabási [5], Dorogovtsev and Mendes [25], and Newman [54], among others. These networks are subjected to both theoretical and empirical analysis to understand their properties. In addition to the other important properties of complex networks like the small-world effect [50], power-law degree distribution [4], and high clustering coefficient, the tendency of nodes in these networks to form community structures, groups of nodes with dense internal connections and sparse connections with the rest of the network, is also studied.

Intuitively, a community is defined as a group of nodes that share dense internal connections (among themselves) and sparse external connections (with the rest of the network). A community could correspond to a functional unit or could be the result of assortative mixing of entities in the underlying system [54]. The community structure of a network is considered overlapping if some nodes in the network simultaneously belong to more than one community. Clustering, in general, is the process of grouping entities based on a similarity measure such that entities within a cluster are similar to each other and entities from different clusters are dissimilar. Community detection or graph clustering is the process of identifying clusters or communities of nodes such that nodes belonging to the same cluster have a lot of in-cluster edges and nodes from different clusters have fewer between-cluster edges [24].

The study of complex networks is a multidisciplinary domain with a rich variety of problems to solve, and one interesting and important problem is the detection of community structures in them. The problem is interesting because even after over a decade of research, we do not have a precise technique that can uncover the community structure in a network absolutely. It is also important because detecting communities in a network helps one understand their structural properties better. Since the structure of a network models the interaction patterns between entities, understanding this in turn helps one uncover any one-to-one mapping between structure of the modules and their functions in the underlying systems. This results in significant insights that have wide applications in information or disease spread, effective immunization, and word-of-mouth marketing, to name a few.

Community detection in complex networks is difficult for a variety of reasons. Communities are not isolated structures, and entities in the system undergo various interactions resulting in a complicated community structure in the network. The difficulty is compounded by the fact that these entities may belong to multiple communities simultaneously. When nodes belong to overlapping regions, they share a good number of edges with all the communities they are assigned to, thereby posing a threat to our definition of communities, which requires sparse external connections of nodes. In addition to this, real-world complex networks may contain millions of nodes and billions of edges, and this renders a number of solutions to these problems computationally intractable. Other challenges here include the lack of good visualization techniques for large graphs and the lack of adequate validation techniques to understand the performance of many overlapping community detection algorithms on real networks.

In spite of the above mentioned difficulties, many overlapping community detection techniques have been developed, motivated by the prospect of valuable insights one can get from the analysis of these networks. One example of this is the analysis of a large Belgian mobile phone network described in Lambiotte et al. [39] and Blondel et al. [13]. Blondel et al. [13] apply their community detection algorithm on this network, and the analysis of communities uncovered by this technique reveals the linguistic homogeneity of people within communities in the bilingual society of Belgium. Such analyses further help in understanding factors that influence social cohesion and the potential fragility of a society. Another well-known example is the analysis carried out on Zachary's Karate Club network from Zachary [76] shown in Figure 1.1. Application of community detection techniques on this network reveals two communities (shown in blue and red). On analysis, these are found to correspond to the two clubs that resulted from the fission of a university karate club. Being able to identify overlapping communities and, in turn, overlapping nodes in networks would uncover a wealth of information about the underlying complex systems. There are many such examples that motivate researchers to focus on the problem of overlapping community detection in complex networks.

## 1.2 Problem

It has been observed that nodes in complex networks belong to multiple communities simultaneously, resulting in an overlapping community structure in the network, as first pointed out by Palla et al. [57]. Detecting overlapping communities adds another dimension to community structure analysis since the detected overlapping nodes could correspond to important or influential entities in the system. There are many techniques proposed for overlapping community detection based on different approaches like link partitioning, local expansion and optimization, clique percolation, and agent-based detection, to name a few. However, the performances of these algorithms as shown in Xie et al. [73] can be significantly improved in detecting both global overlapping community structure as well as individual overlapping nodes and their memberships.

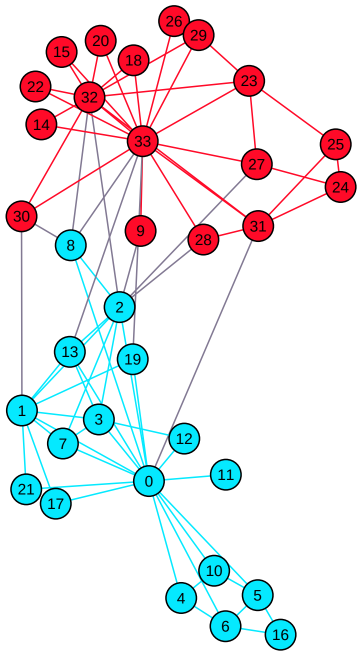


Figure 1.1: Zachary's Karate Club Network showing the two communities in blue and red

Most of these algorithms do not focus on explicitly detecting overlapping nodes in complex networks, and this affects their performance on networks where such nodes belong to many communities (more than 2). As a result, most of the algorithms either overdetect or underdetect the overlapping regions in community structures. There is a need for an algorithm that detects the overlapping community structure in the network as close to the original community structure as possible even when the network contains high overlapping density (many overlapping nodes) and high overlapping diversity (many communities per overlapping node). The algorithm should be able to detect as many overlapping nodes as possible along with their community assignments.

To achieve this, I propose the algorithm IDLE (improved detection using local expansion), which is a local expansion and optimization based technique for overlapping community detection in complex networks. In this, I explore the idea of taking advantage of the general properties of overlapping nodes in a network to better detect the complete overlapping community structure of the network. To begin with, IDLE uses an interesting idea from influential spreader research to identify nodes that can be used as seeds for local community detection. These nodes, whose natural communities are to be constructed, are identified by performing $k$-core decomposition of the original network. The algorithm combines an efficient expansion technique with pruning to identify natural communities of seeds. The key feature that distinguishes IDLE from the other algorithms is that it explicitly focuses on detecting overlapping nodes and their community memberships as it progresses. As a result, it shows improved performance in detecting overlapping communities in both synthetic and real networks. This thesis also presents a systematic study of modularity-based evaluation techniques on real networks, to identify the most relevant measure to report results. This approach has not been explored in earlier studies, and hence it serves as a value addition.

## 1.3 Reader's Guide

The paper is organized as follows. Chapter 2 introduces the preliminaries required to understand the remainder of this thesis. Chapter 3 provides a literature review of the work conducted in the area of complex networks along with a brief survey of the other overlapping community detection techniques. Chapter 4 contains a detailed account of my algorithm IDLE along with pseudocode for different modules. Chapter 5 discusses the experiments conducted using IDLE and two other algorithms that are considered state-of-the-art in overlapping community detection. The results obtained on both synthetic and real networks are also presented here. Chapter 6 concludes the thesis with directions for future work.

# Chapter 2

# Preliminaries

Complex networks are large graphs, $G = (V, E, W)$, whose nodes $V$ represent entities in the system and edges $E$ model associations or interactions. The edges can be directed or undirected, depending on the association between entities: unidirectional associations are represented as directed edges and bidirectional associations as undirected ones. The terms graph and network are interchangeably used in this context. The edges sometimes carry weights $W$ that quantify the strength of the corresponding connections. Most of the analysis techniques that are applied to weighted graphs can also be applied to unweighted ones, since the edges in unweighted graphs are usually assumed to carry a weight equal to 1. IDLE is designed to work on both weighted and unweighted networks.

A clique is a complete subgraph in a network, i.e., a subgraph in which every node is connected to every other node in the subgraph by an edge. A $k$-clique is a clique containing $k$ nodes. A maximal clique is a complete subgraph that is not contained in a larger clique. A maximal clique is also called the maximum clique if it is the largest clique in the given network. Bron and Kerbosch [16] present an algorithm to list all the maximal cliques in an undirected graph. Though there are up to $3^{\frac{n}{3}}$ maximal cliques in a graph with $n$ nodes, as pointed out by Tomita et al. [68], their algorithm is shown to run very fast in practice. Also, finding maximal cliques in a sufficiently sparse graph is scalable, and it does not add an overhead when applied on synthetic and real networks [43].

A community is defined as a set of nodes that are densely connected among themselves and sparsely connected to the rest of the network. Most community detection algorithms for complex networks are focused on detecting non-overlapping communities, resulting in a partition $P$ of $V$ in which each node in the network belongs to only one community, that is,

$$P = \{c_1, c_2, ..., c_k\},$$

where $c_i \cap c_j = \emptyset$ for any two communities $c_i$ and $c_j$ in $P$. Figure 2.1 illustrates the partition of a network into three communities. Each node in the network belongs to a single community and the entire network is parititioned into three communities shown in three different colors.

In the case of overlapping communities, where a node can belong to more than one community, the set of communities found form a cover C, that is,

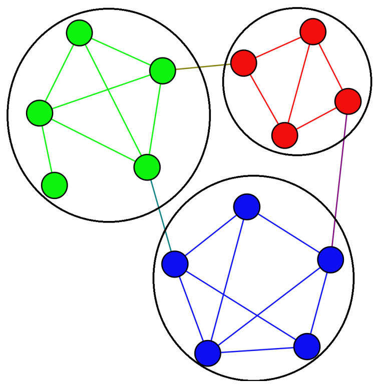


Figure 2.1: Partition of a network showing three communities. The nodes in the different communities are shown in different colors.

$$\mathrm{C} = \{c_1, c_2, ..., c_k\},$$

where any elements of $c_i \cap c_j$ are overlapping nodes. Figure 2.2 illustrates the cover obtained for a network containing three communities. Two communities in this cover overlap with a single overlapping node lying in the overlapping region. This node is shown in black in Figure 2.2.

In a cover, community assignments to nodes can be of one of two types, fuzzy [28] or crisp. A fuzzy assignment involves soft membership, i.e., a node's assignment to a community is associated with a belonging factor that quantifies the strength of this assignment. For each node, the sum of all its belonging factors is equal to 1. In a crisp assignment, the assignments are binary, i.e., a node either belongs to a community or does not. Most overlapping community detection algorithms, including IDLE, produce crisp assignments.

Modularity was introduced by Newman and Girvan [55] as a measure of quality of a network partition. It quantifies community strength by comparing the fraction of edges within the community with such fraction when random connections between the nodes are made. Let $A$ be the adjacency matrix representation of graph $G$. If $G$ is unweighted, $A_{uv}$ contains value 1 if there is an edge between nodes $u$ and $v$, and 0 otherwise. If $G$ is a weighted graph, $A_{uv}$ contains the weight of the edge between nodes $u$ and $v$. Given $A$, the total number or weight of internal edges in a community $c$, $|E_c^{in}|$, is defined as

$$|E_c^{in}| = \sum_{u \in c, v \in c} A_{uv}.$$

The total number or weight of external edges from the nodes in a community $c$, $|E_c^{out}|$, is defined as

$$|E_c^{out}| = \sum_{u \in c, v \notin c} A_{uv}.$$

The total number or weight of edges in the entire network, $|E|$, is straightforward:

$$|E| = \sum_{uv} A_{uv}.$$

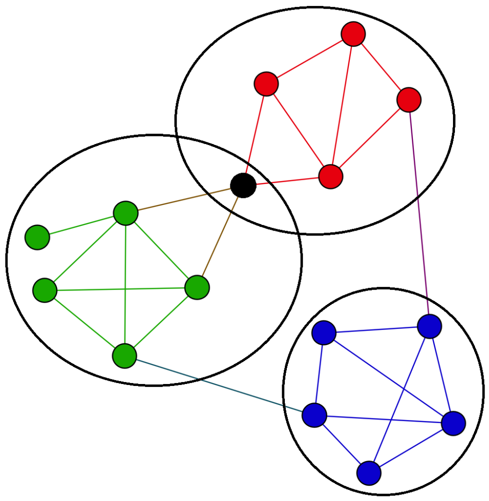


Figure 2.2: Cover obtained for a network containing three communities. The nodes in the different communities are shown in different colors. There is a single overlapping node shown in black that belongs to two communities simultaneously.

Using these, modularity $Q$ of a partition of $G$ is expressed as

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

where $C$ is the set of communities in the partition. There is an equivalent formulation for modularity that is commonly used in literature. For any node $u$ in $G$, its degree is the total number or weight of edges that are incident on it. For any two nodes $u$ and $v$ in $G$, if $c_u$ and $c_v$ are their respective communities in a partition, the Kronecker delta $\delta_{c_u,c_v}$ is equal to 1 if $c_u = c_v$, and 0 otherwise. That is, $\delta_{c_u,c_v}$ is equal to 1 only if both $u$ and $v$ are assigned to the same community. Using these, modularity is also expressed as

$$Q = \frac{1}{2m} \sum\nolimits_{uv} \left[ A_{uv} - \frac{k_u k_v}{2m} \right] \delta_{c_u,c_v},$$

where $m$ is the sum of weights of all edges in the network (which is nothing but $|E|$). Modularity takes values between -1 and 1, and a larger value indicates a better partition of the network.

# Chapter 3

# Related Work

One of the most cited papers in the area of complex networks research is Newman [54]. It is one of the earliest papers that was published when complex networks research was in its nascent stage, and its popularity can be attributed to its exhaustivity as well as its time of publication. Complex networks represent a multidisciplinary area, and Newman [54] distinctly shows the contributions made to this area from researchers in different domains such as biology, physics, social studies, technology, and information. Being a review article, this paper clearly walks the reader through the following: how complex systems are modeled as networks; the components of complex networks; and the properties that are characteristic of such networks. Most of the material in this paper is focused on reviewing the various mathematical models that are developed for complex networks, starting from the simple and elegant Erdos-Renyi random graph models to the more complex and meaningful preferential attachment models. Understanding how these models were developed one after another not only helps the reader understand and appreciate the progress made in complex networks research, but also gives him a better idea about the structure and functions of these networks in general. In addition to the structural models that are discussed, the paper also reviews some of the well-known models for epidemiological processes taking place in complex networks. The paper concludes with a list of significant research problems in the study of complex networks, some of which still remain as open problems. Overall, this paper served as a guide for the fundamentals of complex networks that I needed to understand before starting research in this area.

In complex networks, communities are formed by groups of nodes that correspond to functional units, or those that share some common attributes. Newman [54] introduces the problem of community detection and its importance, but he does not go into details of the complexities involved in community detection in these networks in this paper. The idea behind several community detection techniques is to find a partition of the network such that the size of the cut is minimized. Finding the optimal partition is NP-hard, and hence several appromixation algorithms are proposed for this problem. Spectral partitioning techniques were proposed as a way of approximating the normalized cut or conductance using the normalized Laplacian of a network [65]. In [55], Newman and Girvan introduce a partition quality measure called modularity and discuss betweenness-based approaches for finding partitions of networks. A random walker on complex networks is believed to get "trapped" into densely connected subgraphs corresponding to communities and this intuition has been used in many random walks based community detection techniques [58, 70]. Several other techniques have also been proposed for community detection in literature.

An early paper in the area of community detection is Palla et al. [57], which introduces the problem of detecting overlapping community structures. This paper explains that communities in networks are not necessarily separate subgraphs but can be overlapping cohesive groups of nodes. The authors describe the various studies they had conducted, which led to their understanding of the nature of overlapping communities in complex networks. To solve the problem of detecting overlapping communities, the authors introduce terms such as membership number, overlap size, and community degree. Using these, they propose a clique percolation based algorithm to detect overlapping communities. The algorithm produces promising results on small graphs with overlapping community structures in them. However, the exponential time complexity of their algorithm forbids its application to complex networks of hundreds of millions of nodes and billions of edges.

The concept of overlapping community structures in turn opens up an array of related problems in complex networks. Further research in this area revealed that networks often show a hierarchical organization, with communities embedded within other communities. To detect communities well, we need an algorithm that can efficiently detect both overlapping and hierarchical community structures in complex networks. Lancichinetti et al. [41] presents the first algorithm that does this. Similar to the idea of Baumes et al. [12], the basic assumption behind this algorithm is that communities are essentially local structures that are built around sets of seed nodes, resulting in overlapping subgraphs. The authors propose a local community detection technique based on the maximization of a fitness function with tunable parameters. This fitness function determines whether a node should be included in the current local community that is being explored, and the tunable parameters determine the scale of detected local communities. The efficiency of this algorithm is dependent on the sparsity of large networks, where each node is not connected to most of the other nodes in the network. The authors reveal that it is enough to select a few random seed nodes in the graph and determine their natural communities. Putting these natural communities together in turn results in the required overlapping community structure of the network. This paper also extends the NMI (normalized mutual information) to compare the built-in modular structure of networks with the results of overlapping community detection algorithms. I have used this NMI as one of the measures to capture the performance of different algorithms in Chapter 5.

An overlapping community detection technique that is very close to [41] is introduced in Lee et al. [43]. This technique, called GCE (greedy clique expansion), uses the nodes in maximal cliques of size at least $k$ as seed nodes. The seeds are then subject to a greedy expansion resulting in local communities. Lee et al. [43] use the same fitness function as [41] for greedy expansion. However, they improve the performance of GCE by optimally removing duplicate communities. They also employ other useful optimization techniques that improve the efficiency of their method. Though this method is observed to perform well on many networks containing hidden overlapping communities, its performance deteriorates when there is an increase in the degrees of nodes.

An extensive and most recent survey of overlapping community detection techniques is conducted by Xie et al. [73]. Fourteen different overlapping community detection techniques of different types, such as clique percolation, link partitioning, local expansion and optimization, fuzzy detection, and agent-based detection, are evaluated on real and synthetic benchmark graphs. Their performance is measured using NMI and Omega index [21], and an overall final rank is determined. The evaluation techniques in this paper can be used to compare the performance of any new overlapping community detection algorithm with the existing ones. I have conducted experiments similar to the ones presented in [73] to compare the performance of IDLE with two of the current state-of-the-art algorithms.

Another important problem in the study of complex networks is the identification of influential spreaders. Influential spreaders are nodes that play an important role in the propagation of information or diseases in these networks. Newman [54] introduced some models used in the study of epidemiological processes in complex networks, but it did not discuss influential spreaders. Kitsak et al. [35] is an important milestone in the research of influential spreaders. It is generally believed that the most connected people are the key influential spreaders in a network. But this paper proves that the topology of the network organization plays an important role and that the nodes in the core of the network identified by $k$-core decomposition analysis are the most efficient spreaders. The authors conduct experiments to compare spreading, when seed nodes are high degree nodes versus high $k$-core nodes, and present experimental results to clearly show that spreading is increased when one starts from the high $k$-core nodes. Since these nodes form the core of the network, they also result in better initial communities when used as seed nodes in IDLE and, in turn, an efficient coverage of the network.

Blondel et al. [13] introduce a heuristic-based modularity optimization algorithm, now called the Louvain method, for efficient community detection in complex networks. Modularity is an important measure that is often used to gauge the quality of partitions resulting from community detection techniques. The Louvain method is a two-pass algorithm to determine community structure in large networks, based on modularity optimization. Though this algorithm does not detect overlapping communities, it is faster than all other known community detection methods in terms of computation time. It first enabled the algorithmic analysis of complex networks of magnitudes that could not be analyzed earlier due to computational costs. The authors also describe the application of this algorithm to a large mobile communication network of over 2 million nodes and 38 million edges, to convey how the analysis of such large networks becomes feasible with the use of their algorithm. The Louvain method is one of the fastest known techniques for disjoint community detection in complex networks, and I have used the Louvain partition of networks to compare results in my experiments in Chapter 5.

# Chapter 4

# Improved Detection Using Local Expansion

There have been many methods proposed for detecting overlapping communities in complex networks. Each of these methods falls under one of the broad categories as discussed in Chapter 3. From a survey of 14 of these algorithms conducted by Xie et al. [73], we see that there is room for significant improvement in the quality of detection, especially when the difficulty of the detection task increases (i.e., when the overlapping nodes belong to a large number of communities). As a consequence, most of these techniques do not accurately detect all the overlapping nodes in the network. In an attempt to improve detection capability, I propose a method for better and faster overlapping community detection called IDLE (improved detection using local expansion). As the name suggests, IDLE is a local expansion and optimization based technique for community detection. It is primarily designed for undirected networks and can be applied to both weighted and unweighted networks. In this chapter, I discuss different components of the proposed method, along with details about their complexities and optimizations wherever applicable. I supplement these sections with relevant pseudocode for the modules.

## 4.1 Seeds Selection

Since IDLE is a local expansion technique, the first step is selection of seeds that will eventually be used to construct local communities. As observed by some local expansion techniques discussed in Chapter 3, I believe that the quality of communities detected depends on the quality of seeds selected for local expansion. This is true because the final overlapping community structure that the algorithm develops depends on the natural communities of these selected seeds. To ensure that the seeds we select for local expansion result in meaningful communities, I adopt an idea from influential spreader research.

From Kitsak et al. [35], we know that nodes in the core of the network, identified using the $k$-core decomposition technique, are the most influential spreaders. Our approach is based on the hypothesis that these nodes will also serve as the best seeds for overlapping community detection. In IDLE, we first perform a $k$-core decomposition of the graph. This $k$-core decomposition is an elegant technique, first introduced in Seidman [62], that reveals the structure of a graph based on the connectivity of nodes and their degrees. Given a graph $G = (V, E)$, we start with the minimum possible value for $k$ and iteratively delete all the nodes of degree less than $k$ along with their incident edges. The remaining graph is the $k$-core. We continue this for increasing values of $k$ until no nodes remain in the graph. At the end of the procedure, the nodes in the network are assigned to their $k$-cores based on degree and structure, and nodes that belong to the highest $k$-core form the main core of the graph. The properties of these cores along with an efficient algorithm to perform $k$-core decomposition of a graph is discussed in Batagelj and Zaversnik [11]. All nodes in a given $k$-core have degree greater than or equal to $k$. In general, the $k$-cores are larger for small values of $k$, and they diminish in size as the value of $k$ increases.

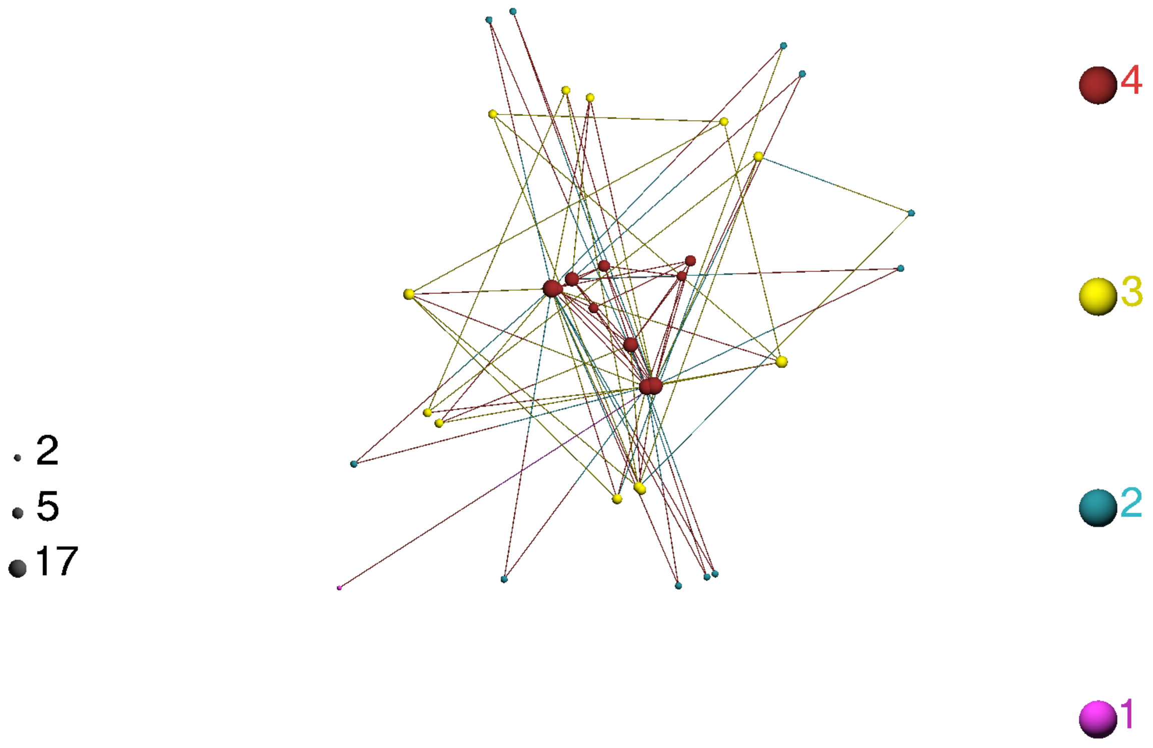


Figure 4.1: The result of applying $k$-core decomposition on Zachary's Karate Club Network. The legend on the right shows the colors corresponding to different $k$ values. On the left, the range of degrees of the nodes is shown. This visualization was created using the LaNet-vi tool [1, 6].

Figure 4.1 shows the result of $k$-core decomposition performed on Zachary's Karate Club network [76]. The actual network with its ground-truth communities is shown in Figure 1.1. Here, dark colored nodes (corresponding to $k$=4) constitute the main core of the network. The size of the nodes represent their degrees in the graph. We see that the high $k$-core nodes are also the ones with higher degrees in this case. Table 4.1 shows the nodes (identified by node numbers used in Figure 1.1) in the karate club network grouped according to their corresponding $k$-core values and degrees. This gives the reader an idea about the relation between degrees and $k$-core values of nodes, the relative size of $k$-cores in the network, and the attributes of the most influential nodes (node 0 and node 33) in this network.

Table 4.1: Table showing the nodes in Zachary's Karate Club network grouped according to their k-core values and degrees

| $k$ | 4 | | | | | | | | 3 | | | | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Degree* | 17 | 16 | 12 | 10 | 9 | 6 | 5 | 4 | 6 | 5 | 4 | 3 | 2 | 1 |
| *Nodes* | 33 | 0 | 32 | 2 | 1 | 3 | 8,13 | 7,30 | 31 | 23 | 5,6,27,29 | 4,10,19, 24,25,28 | 9,12,14, 15,16,17, 18,20,21, 22,26 | 11 |

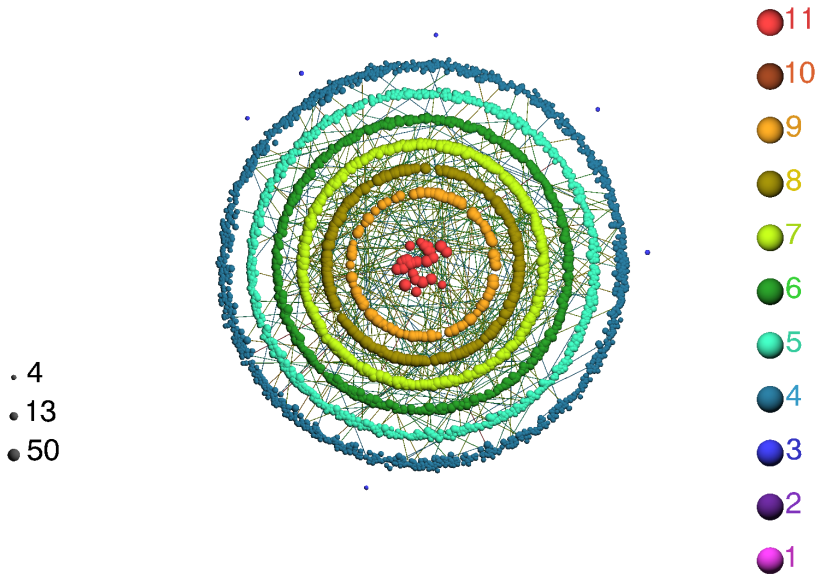


Figure 4.2: The result of applying $k$-core decomposition on an LFR benchmark network, defined in Section 5.1.1, containing 5000 nodes. The legend on the right shows the colors corresponding to different $k$ values. On the left, the range of degrees of the nodes is shown. This visualization was created using the LaNet-vi tool [1, 6].

Figure 4.2 shows the result of $k$-core decomposition performed on an LFR benchmark graph [42] using the Large Networks Visualization tool [1, 6]. This is a relatively large network with 5000 nodes containing overlapping communities of size ranging from 10 to 50 nodes each. Each ring shown in the figure corresponds to a different $k$-core of the network. As mentioned, we see that the size of $k$-cores decreases with increasing values of $k$. From the distribution of the sizes of nodes in

Figure 4.2, we also observe that nodes with high degrees are distributed in different $k$-cores. Figure 4.3 shows the pseudocode for $k$-core decomposition, in its naive implementation, to give the reader an overview of the technique. There is a more efficient implementation for $k$-core decomposition discussed in Batagelj and Zaversnik [11], and we use this version in IDLE.

**Figure 4.3** Algorithm $K$-CORE DECOMPOSITION

```
Input: G(V, E), graph representation of the complex network with $|V|$ nodes and $|E|$ edges
Output: core, an array containing the $k$-core value for each node, in its corresponding index
 1: for i ← 1 to $|V|$ do
 2:     $degree[i]$ ← degree of node i in G
 3:     $core[i] \leftarrow -1$                    ▷ Initialize $k$-core value for all nodes to -1
 4: end for
 5: $count \leftarrow |V|$
 6: $k \leftarrow$ MIN($degree[\,]$)               ▷ The least possible value for $k$ is the lowest degree
 7: while count > 0 do
 8:     $flag \leftarrow 0$
 9:     for each node $v \in V$ do
10:         if $degree[v] \leq k$ and $core[v]$ == -1 then    ▷ If node has degree $\leq k$ and has no $k$-core value
11:             $core[v] \leftarrow k$
12:             for each node $u \in Neighbors(v)$ do
13:                 if $core[u]$ == -1 then
14:                     $degree[u] \leftarrow degree[u]$ - 1
15:                 end if
16:             end for
17:             $count$ ← count - 1            ▷ One more node has been assigned its $k$-core value
18:             $flag \leftarrow 1$
19:         end if
20:     end for
21:     if $flag == 0$ then
22:         $k \leftarrow k + 1$               ▷ If no more nodes can belong to current $k$, increment $k$ by 1
23:     end if
24: end while
25: return $core$
```

At the end of $k$-core decomposition of the network, each node has an associated $k$-core value. IDLE starts by picking seeds, one at a time, for local expansion, starting from the highest $k$-core arranged in decreasing order of degrees. We sort nodes in this order, because this helps in arranging nodes in decreasing order of influence within a given $k$-core. It makes sure that the next seed that is selected is not contained in any of the communities constructed earlier. This helps in handling the issue of removing duplicate communities discussed in Lee et al. [43] better, and this also contributes to the efficiency of IDLE. It is possible that there are some nodes that are still not assigned to any communities after we have expanded qualifying seeds in the main core. To handle such cases, IDLE repeats this seed selection and expansion process on nodes in lower $k$-cores until all the nodes in the graph are covered in at least one local community. This idea of picking seeds from the lower $k$-cores is justified because we cannot assume all communities in the network have the same size and the same internal structure. This idea helps us to detect communities of all scales and contributes to the accuracy of the results. Like other local expansion techniques, IDLE also does not require a user to specify the number of communities in the network in advance.

In terms of complexity, the seed selection process involves two major steps: sorting nodes in decreasing order of degrees and $k$-core decomposition. In [11], Batagelj and Zaversnik present an efficient algorithm with complexity $O(|E|)$ for $k$-core decomposition of a graph. Therefore, the worst case time complexity of this step is $O(|V| \log |V|)$ for sorting plus $O(|E|)$ for $k$-core decomposition and the selection of seeds from $V$ nodes in the network.

## 4.2 Fitness Function

The next step after seed selection is local expansion where IDLE constructs a community around the seed node. Before I discuss the process of local expansion, I would like to introduce the fitness function [41] that plays an integral role in this step. Local expansion is usually carried out by optimizing an objective function that reflects the community property: dense internal connections (between nodes in the community) and sparse external connections (with nodes in the rest of the network). A fitness function is also referred to as a weight function [12] or local density function. Such a fitness function is designed to measure the goodness of communities, and one aims at either minimizing or maximizing it in the process of detecting good communities.

Conductance is one of the earliest [66] and widely adopted [48, 33] measures for quality of communities. It is also called the normalized cut metric [65]. Consider a graph $G = (V, E, W)$ with adjacency matrix $A$. In the case of an unweighted graph, $A_{uv}$ contains 1 if there is an edge between nodes $u$ and $v$, and 0 otherwise. In the case of weighted networks, $A_{uv}$ contains the weight of the edge between $u$ and $v$, if it exists, and 0 otherwise. As we know, $V$ and $E$ represent the set of nodes and edges in the graph respectively, and $W$ is the set of weights of the edges in $E$. Consider a set $S \subset V$. The volume of set $S$ is defined as

$$vol(S) = vol(S, V) = \sum_{u \in S} \sum_{v \in V} A_{uv}.$$

That is, the volume of a set equals the sum of weighted degrees of nodes in the set. In other words, it is equal to the sum of weights of external edges plus twice the total weight of all the internal edges attached to the nodes in it. This idea can be extended to find the volume between two sets $S_1$ and $S_2$ ($S_1 \subset V$ and $S_2 \subset V$) by substituting $S_1$ and $S_2$ for $S$ and $V$. The volume between sets $S_1$ and $S_2$ is defined as

$$vol(S_1, S_2) = \sum_{u \in S_1} \sum_{v \in S_2} A_{uv}.$$

From its definition, we understand that this measures the total weight of edges with one end in set $S_1$ and the other end in set $S_2$. Assuming that S is a community identified in the network, and $\bar{S}$ is the set of nodes not present in $S$ but in $V$, $vol(S, \bar{S})$ is the total weight of external connections of the community $S$. Lucas et al. [33] mention this as the surface area of the boundary between a community and the rest of the network. Using these, the conductance of a set $S$ is defined to be

$$Conductance, \phi(S) = \frac{vol(S, \bar{S})}{min(vol(S), vol(\bar{S}))}.$$

Conductance measures the ratio of the capacity of external connections of a community to the total capacity of nodes in that community. For a good community, we expect external connections to be much sparser than the internal ones. Hence, the lower the value of its conductance, the better is the quality of the community. Conductance is closely related to edge expansion and approximation algorithms for minimizing these quantities are discussed by Leighton and Rao [44] and Arora et al. [7].

Many local expansion techniques [12, 41, 43, 29] use the following fitness function $F$ or its variant:

$$F(S) = \frac{k_{in}^S}{(k_{in}^S + k_{out}^S)^\rho}.$$

Here, $k_{in}^S$ is equal to twice the sum of weights of edges between nodes in the community, and $k_{out}^S$ is the sum of weights of edges with one end in the community and the other end in the rest of the network. The quantities $k_{in}^S$ and $k_{out}^S$ are also called the total internal and external degrees of the nodes of the community respectively. The value $\rho$ is a positive real-valued number called the resolution parameter. By varying the value of $\rho$, one can explore communities at different scales, thereby resulting in a hierarchical community structure. The function $F$ measures the ratio of the total internal degrees of a community to the total degrees of nodes in that community. For a good community, we expect internal connections to be much denser than the external ones. Hence, the higher the value of this fitness function, the better the communities. Results from the comparative study in [73] suggest that a greedy local optimization of this fitness function produces good communities, and it can be verified from the high ranks assigned to these techniques. In this thesis, I focus on detecting overlapping communities in the networks at a natural scale. This corresponds to using the default recommended value for $\rho$, which is 1. Hence the fitness function is simply

$$f(S) = \frac{k_{in}^S}{k_{in}^S + k_{out}^S}.$$

We use $F(S)$ and $f(S)$ to differentiate the two in our discussions.

My initial choice for a fitness function was conductance, because of its already proved usefulness validated by its wide adoption. During experiments, I noted the complementary nature of $\phi(S)$ and $f(S)$. Due to the inherent sparsity of real-world complex networks, nodes in this network will not be directly connected to the majority of them in the rest of the network. Hence one can safely assume that the volume of a community $S$ is smaller than the volume of the rest of the network $\bar{S}$.

Therefore, assuming $vol(S) \leq vol(\bar{S})$,

$$\phi(S) = \frac{vol(S, \bar{S})}{vol(S)}.$$

Since $vol(S)$ is the the sum of weighted degrees of nodes in the set, it can be expressed as the sum of internal and external degrees of nodes in it. This implies

$$vol(S) = k_{in}^S + k_{out}^S.$$

Also, $vol(S, \bar{S})$ is nothing but the total external degrees of nodes in S, that is,

$$vol(S, \bar{S}) = k_{out}^S.$$

Putting together the new expressions for volumes involved, we obtain

$$\phi(S) = \frac{k_{out}^S}{k_{in}^S + k_{out}^S}$$

$$\phi(S) = 1 - \frac{k_{in}^S}{k_{in}^S + k_{out}^S}$$

$$\phi(S) = 1 - f(S).$$

Hence, maximizing conductance is equivalent to minimizing $f(S)$.

The community that gives the maximum value for $f(S)$ gives the minimum value for $\phi(S)$. This implies that when we start from the same seed, maximizing $f(S)$ and minimizing $\phi(S)$ result in the same community. Though IDLE does not focus on detecting hierarchical community structure at this point, I believe that it will definitely be considered for future work. Since the addition of the resolution parameter to $f(S)$ results in $F(S)$ that has been well explored, this serves as an added advantage when one considers multi-scale overlapping community detection. Keeping this in mind, the fitness function I choose for local expansion in IDLE is

$$f(S) = \frac{k_{in}^S}{k_{in}^S + k_{out}^S}.$$

## 4.3 Local Expansion

Once a seed has been selected and IDLE has ensured that it has not been covered in any of the communities built so far, the next step is to perform local expansion around this seed. Local expansion is the process of building the *natural community* of a given seed. As discussed in Section 4.2, this is achieved by putting nodes from the neighborhood that optimize a fitness function into the community. In this section, I discuss the steps involved in performing a local expansion.

Before the details of local expansion are described, I introduce the 1-neighborhood of a community. The is also referred to as its frontier. The 1-neighborhood of a community is the set of nodes that do not belong to the community, but have an edge from the nodes in the community. This 1-neighborhood can be visualized as a hypothetical band around a community containing all the nodes that have any connections with the nodes inside the community. For every seed node that is selected, IDLE constructs its initial community. The initial community of a seed is constructed as follows: IDLE finds the subgraph, say $H$, induced by the seed node $s$ along with its neighbors; it lists the maximal cliques contained in $H$ using algorithms discussed in Bron and Kerbosch [16] and Tomita et al. [68]; it then returns the largest maximal clique from this list as the initial community. IDLE then computes the value of fitness of this initial community, along with its 1-neighborhood. A 1-neighborhood reveals an important property of the selected seed node that will be discussed later in this section.

IDLE proceeds to perform a greedy maximization of the fitness function $f(S)$ discussed in Section 4.2 by selecting a single node $v$ from the 1-neighborhood that causes the maximum increase in $f(S)$. A series of steps follow: the selected node $v$ is added to the community; since $v$ now belongs to the community, it is removed from the 1-neighborhood of the community; the 1-neighborhood is further updated by adding the neighbors of $v$ to it. All these result in an expansion of the local community and the corresponding 1-neighborhood. IDLE repeats this process until the point when the addition of a new node to the community does not cause an increase in the fitness function. At this point, it stops local expansion and proceeds to prune the newly formed community. This process of pruning will be discussed in detail in Section 4.4.

I would like to discuss an important feature of IDLE that separates it from the other overlapping community detection techniques and results in its improved community detection capability. A node that is selected as seed could be of one of two types: a non-overlapping node that belongs to a single community or an overlapping node lying in the region of overlap of two or more communities. IDLE understands and handles these cases efficiently, and this is one of the prime reasons for its better performance. The 1-neighborhood of a selected seed helps IDLE differentiate seeds of these two kinds. If a seed is a node that belongs to the overlapping region, it could have one or both of the following properties:

1. The neighbors of this node do not all belong to a single community, and when one observes the proportions of neighbors belonging to different communities, there is not a single community that holds the majority of these neighbors. That is, this node is in turn connected to multiple communities through its edges.
2. Most of the communities of the neighbors of this node do not choose to include it. This is because of the way the edges of this node are distributed, satisfying property 1. The addition of this node does not cause an increase in a community's fitness function because the majority of its edges will always be external connections, while $f(S)$ favors a node that has denser internal connections and sparser external connections.

In local expansion, IDLE mainly uses property 2 to differentiate between types of seeds, as a node that is strongly connected to a single community would not have these properties. From experiments, I observe that using a node from the overlapping region as seed results in the addition of nodes from its 1-neighborhood that do not all belong to the same community in the ground-truth community structure of the network. This is true, because local expansion initially includes neighbors that all cause a slight increase in the fitness function (and belong to different ground-truth communities), before it starts gaining nodes that map to a single ground-truth community. Once this majority is achieved, more nodes from their ground-truth community are included until the fitness function cannot be improved any further. This phenomenon affects the performance of local expansion based overlapping community detection algorithms. Hence it is important to detect the type of the seed used for local expansion.

To achieve this, IDLE observes the proportion of neighbors in the 1-neighborhood of a seed node that has already been assigned to communities. It chooses a threshold $p$ for this proportion. If the observed proportion of neighbors that are already assigned communities is above the threshold $p$, IDLE immediately identifies this node to belong to the overlapping region and does not proceed with local expansion. If not, the seed is assumed to belong to a single community and local expansion follows. The size of the 1-neighborhood of a seed is used to determine the value for $p$. If the size is small (i.e., size $\leq 4$), IDLE uses the value of $p = 0.5$. For any other size of 1-neighborhood (i.e., size $> 4$), it uses $p = 0.8$. These values for $p$ are observed to produce good results from experiments, and they can be changed if properties of the complex networks are known in advance.

The time complexity of this step cannot be determined in advance since the expansion (and in turn the number of steps) depends on the structure of the graph. However, I have implemented simple optimizations that speed up community expansion. This is discussed in detail in Section 4.6. Figure 4.4 presents the pseudocode for local expansion implemented in IDLE.

## 4.4 Pruning

Pruning is an important process that is executed as part of local expansion in IDLE. From the results of experiments, I observe that this step also plays an important role in the objective of IDLE to explicitly detect overlapping nodes and in turn improve overlapping community detection. As discussed in the previous sections, seed selection is followed by greedy maximization of a fitness function that results in local expansion. Once the natural community of a given seed $s$ is determined, the community is subject to pruning. There are two essential operations performed as part of this:

1. Removal of non-core nodes, and
2. Detection and removal of near-duplicate communities.

Removal of nodes is performed in other local expansion and optimization techniques [12, 41] at different points in their algorithms, for different reasons. In this research, I observe that when IDLE begins with local expansion for a seed, certain nodes from its 1-neighborhood that do not strongly belong to the natural community of the seed might be added due to a slight increase in the fitness function. These nodes are considered to be non-core nodes (i.e., weakly embedded in the community) and should be pruned from the community. During early stages of seed selection, some overlapping nodes may be selected as seeds from high $k$-cores due to the rich connectivity of nodes in the overlapping region. If not many of its neighbors are covered yet at this point, IDLE will not be able to filter out this overlapping node using property 2 discussed in Section 4.3. Such a community requires pruning to closely map to a ground-truth community in the network. Due to the way $f(S)$ is defined, strategically less advantageous nodes sometimes result in a small increase in the fitness value, resulting in the addition of such nodes. This is another instance where pruning becomes essential. I have explored two approaches for pruning, each of which comes with its own advantages.

The first approach is based on the strength of connection of a node to a community. The internal degree of a node $v$ with respect to community $c$, $d_{vc}^{in}$, is the sum of weights of all edges connecting $v$ to other nodes that belong to $c$. It is defined as

$$d_{vc}^{in} = \sum_{u \in c} A_{uv}.$$

Similarly, the external degree of a node $v$ with respect to community $c$, $d_{vc}^{out}$, is the sum of weights

**Figure 4.4** Algorithm LOCAL-EXPANSION

```
Input: G(V, E), graph representation of the complex network,
    s, the selected seed,
    Covered[|V|], a boolean array that contains 1 if a node is already covered 0, if not.
    Cover, the list of communities constructed so far.
Output: C, a list containing nodes that represent the local community of seed s
 1: $N \leftarrow$ 1-NEIGHBORHOOD($s$)
 2: $count \leftarrow 0$
 3: for each node $n \in N$ do
 4:     if $Covered[n] == 1$ then                    ▷ If node has already been covered
 5:         $count \leftarrow count + 1$
 6:     end if
 7: end for
 8: $ratio \leftarrow \frac{count}{\text{SIZE}(N)}$
 9: if (SIZE($N$) $\leq 4$ and $ratio > 0.5$) or (SIZE($N$) $> 4$ and $ratio \geq$ p) then
10:     return [ ]            ▷ Return empty list to indicate seed is an overlapping node
11: end if
12: $H \leftarrow$ SUBGRAPH($G, s$)           ▷ H is the subgraph induced by s and its 1-neighborhood
13: $C \leftarrow$ LARGEST-MAXIMAL-CLIQUE($H$)  ▷ Initial community is the largest maximal clique in H
14: $N \leftarrow$ 1-NEIGHBORHOOD($C$)               ▷ Get the inital 1-neighborhood of community
15: $flag \leftarrow 1$
16: $CurrentFitness \leftarrow$ GET-FITNESS($C$)
17: $InternalDegree \leftarrow$ GET-INTERNALDEGREE($C$)
18: $ExternalDegree \leftarrow$ GET-EXTERNALDEGREE($C$)
19: while flag == 1 do
20:     $MaxFitness \leftarrow CurrentFitness$
21:     $flag \leftarrow 0$
22:     for each node $v \in N$ do
23:         $k\_in \leftarrow InternalDegree + 2 * \text{G.WEIGHT}(C, v)$
24:         $k\_out \leftarrow ExternalDegree - \text{G.WEIGHT}(C, v) + \text{G.WEIGHT}(V - C, v)$
25:         $f \leftarrow \frac{k\_in}{k\_in + k\_out}$
26:         if $f \geq MaxFitness$ then        ▷ If computed fitness is greater than current max
27:             $MaxFitness \leftarrow f$
28:             $Selected \leftarrow v$
29:         end if
30:     end for
31:     if $MaxFitness > CurrentFitness$ then
32:         C.ADD($Selected$)
33:         $CurrentFitness \leftarrow MaxFitness$
34:         $N \leftarrow$ 1-NEIGHBORHOOD($C$)
35:         UPDATE($InternalDegree, Selected$), UPDATE($ExternalDegree, Selected$)
36:         $flag \leftarrow 1$
37:     end if
38: end while
39: $C \leftarrow$ PRUNE($G, C, Cover$)
40: return C
```

of all edges connecting $v$ with the remainder of the network. It is given as

$$d_{vc}^{out} = \sum_{u \notin c} A_{uv}.$$

*Community Degree Ratio* of a node $v$ with respect to community $c$ is a straightforward value that is defined as the ratio of internal degree to external degree of the node with respect to that community,

$$\textit{Community Degree Ratio}(v,\ c) = \frac{d_{vc}^{in}}{d_{vc}^{out}}.$$

This approach achieves pruning of a community by iteratively looking at the nodes in it, and removing the ones whose *Community Degree Ratio* values are below a threshold $w$. When IDLE removes a node from the community, it simultaneously updates the degrees of its neighbors that also belong to this community, in turn affecting their community degree ratio values. The value for threshold $w$ is set as 0.3 or 0.4 in my experiments using IDLE. If community degree ratio(v, c) $\leq w$ for any node in the community, it implies that the node has more than preferred external edges and hence it is removed. Because this step updates the degrees of the neighbors of the node belonging to $c$, it may result in further pruning. IDLE performs this iteratively until there is no node left in the community whose community degree ratio falls below the threshold. The other approach for pruning removes nodes whose removal from the community $c$ results in an increase in the fitness value of $c$. As in the previous approach, once a node is removed from $c$, the degrees of its neighbors in $c$ are also updated resulting in further pruning. This can be performed iteratively until there is no node left in $c$ whose removal results in an increase in the fitness of the community.

The first approach is sensitive to the value set for threshold $w$ for community degree ratio and it might result in pruning of entire communities if the threshold set is high and the overlapping density and diversity of the network also are reasonably high. Due to this, it is important to pay attention to the threshold selected and this is usually decided based on the average fitness of communities formed in the network as a result of local expansion. Selecting a threshold that is lower than the average fitness of local communities has been observed to work well in pruning weakly connected nodes from communities. Since the second approach is based on the fitness values of communities themselves and not on a hard threshold, it does not result in over-pruning of nodes. However, we cannot control pruning in this case as we could in the first approach. Keeping in mind advantages and disadvantages of both the approaches, I have employed the community degree ratio based approach in the experiments discussed in Chapter 5.

An important issue that removal of nodes from a community can give rise to, whether it is carried out during each iteration of local expansion for a seed, as it is performed in some overlapping community detection algorithms like IS [12] and LFM [41], or after all the iterations are done, as it is performed in IDLE, is that a community can end up having disconnected sets of vertices. This is not desired of a community which is expected to be a group of tightly connected nodes. This problem, along with a solution to address it, has been discussed in detail by Kelley [34]. In IDLE, I have addressed this problem by observing the connectivity of the community at the end of pruning. If the community contains more than one connected component, the largest connected component is returned as the community from the current local expansion step. Removing nodes at the end of local expansion, instead of during every iteration in expansion, is beneficial in this sense because it is enough to check the connectivity of the community once rather than at every iteration as suggested in [34].

The next operation that is performed as part of pruning is the removal of near-duplicate communities. In GCE [43], removal of duplicate communities is performed by computing a minimum community distance parameter between a candidate community and an optimal list of already accepted communities. If the minimum community distance parameter of the candidate community lies within a threshold $\epsilon$ for any of the accepted communities, then a possible duplication is detected and the candidate community is removed. In IDLE, similarity of the candidate community to an already accepted community is computed using the overlap coefficient (or the Szymkiewicz-Simpson coefficient). For any two communities $c_i$ and $c_j$ in cover $C$,

$$\text{Overlap coefficient} = \frac{|c_i \cap c_j|}{min(|c_i|, |c_j|)}.$$

IDLE performs this by calculating the *count* of nodes in the candidate community, say $c_i$, that are already assigned to an accepted community, $c_j$ this case. It then computes a ratio of this *count* to *size* of the smaller of the two communities $c_i$ and $c_j$. If the overlap coefficient exceeds a threshold $\epsilon$, then the candidate community is removed. The value for this threshold can be set by a network analyst who knows some properties of the network or to a value obtained from experiments. Though the list of accepted communities gets larger as IDLE proceeds, making this operation expensive, we can perform this efficiently by maintaining a dynamic table that maps nodes to their communities assigned so far. The list of accepted communities that are relevant for duplicate detection are only those that contain the nodes that are also assigned to the candidate community. Using the table, this list of relevant accepted communities can be computed efficiently. The pseudocode for pruning is shown in Figure 4.5.

## 4.5 Post-Processing

Most of the local expansion techniques discussed in [73] allow natural communities of seeds to expand and report those nodes that have been covered in multiple communities as overlapping nodes. But the actual overlapping nodes might not be covered in most of the overlapping ground-truth communities they belong to due to the effect of property 2 discussed in Section 4.3. This is an important reason that contributes to the poor performance of many algorithms in identifying overlapping nodes in the network. One can see this from the results presented in Chapter 5 as well as in [73]. IDLE chooses to handle the problem of identifying overlapping nodes by looking for them explicitly, and I observe that this significantly improves its performance (in terms of the F1 score). IDLE detects overlapping nodes in two phases:

1. A mandatory operation, by filtering out seeds as overlapping nodes during local expansion of seeds.

2. An optional post-processing step, by identifying nodes that satisfy property 1 discussed in Section 4.3, after the completion of seed selection and local expansion phases.

There is an important parameter of the network that determines community assignment for overlapping nodes detected. In IDLE, I represent this as $\gamma$. The value $\gamma$ is the absolute count or proportion of neighbors of a node that should belong to a community before the node itself can

**Figure 4.5** Algorithm PRUNE

**Input:** **G(V, E)**, graph representation of the complex network,
**C**, the community obtained after greedy optimization of fitness function,
**Cover**, the list of communities constructed so far,
**Output:** **C**, the final community obtained after pruning

```
 1: $flag \leftarrow 1$                                              ▷ Removal of unwanted nodes begins
 2: while flag == 1 do
 3:     $flag \leftarrow 0$
 4:     for each node $v \in C$ do
 5:         $d_{vc}^{in} \leftarrow$ GET-INTERNAL-DEGREE$(C, v)$
 6:         $d_{vc}^{out} \leftarrow$ GET-EXTERNAL-DEGREE$(C, v)$
 7:         if $d_{vc}^{in} < d_{vc}^{out} \wedge \frac{d_{vc}^{in}}{d_{vc}^{out}} \leq$ w then
 8:             C.REMOVE$(v)$
 9:             $flag \leftarrow 1$
10:         end if
11:     end for
12: end while                                                        ▷ Removal of unwanted nodes ends
13: $C_{Added} \leftarrow$ RELEVANT-COMMUNITIES$(Cover, C)$    ▷ Removal of duplicate community begins
14: for each community $c_i \in C_{Added}$ do
15:     overlap-coefficient $\leftarrow \frac{|c_i \cap C|}{min(|c_i|, |C|)}$
16:     if overlap-coefficient $\geq \epsilon$ then
17:         return [ ]
18:     end if
19: end for                                                     ▷ Removal of duplicate community ends
20: if $|C| \leq 2$ then        ▷ If size of community less than or equal to 2, set C to an empty set
21:     $C \leftarrow [\,]$
22: end if
23: return C
```

be assigned to that community. In the case of weighted networks, this could be a proportion of weighted-degree of the node. This property is a characteristic of the network and might greatly differ from one network to another depending on what the nodes and edges represent.

At the end of local expansion, IDLE has identified a subset of the actual overlapping nodes. It ensures that at the end of seed selection and local expansion, all the nodes in the network are accounted for in at least one of the local communities that are built. To achieve this, it is important to assign the identified overlapping nodes to their communities. If a value for $\gamma$ is specified by the user, IDLE utilizes this to detect communities of the identified overlapping nodes. If not, it chooses a default value of 2 for the count of neighbors. There could be nodes in the network where each of its neighbors belong to a different community. When IDLE identifies such nodes as overlapping, the default value used is 1. That is, if a node has at least 1 or 2 of its neighbors in a community then it is assigned to that community. As a result, the overlapping node is assigned to all the communities of neighbors that satisfy this property. This ensures that IDLE assigns all the nodes in the network to at least one of the communities detected.

IDLE could be modified to perform a post-processing step that primarily involves identification of overlapping nodes, because not all overlapping nodes will be selected as seeds during the seed selection phase, and it is a good idea to identify those overlapping nodes that are missed during local expansion. The detection performed as part of post-processing is relaxed in that IDLE does not assign detected nodes to more communities unless a $\gamma$ value is specified. IDLE examines every node in the network and observes its neighbors. If a majority (some proportion) of its neighbors belong to a single community in the cover or if there is only a single community that holds $\gamma$ of its neighbors in the cover, then it is identified as a non-overlapping node. If not, then this node satisfies property 1 from Section 4.3 and hence will be identified as an overlapping node. The idea behind post processing is that even if IDLE did not identify this node to be an overlapping node during the earlier steps, it will be detected at this point. If a $\gamma$ value is specified, then community re-assignment is performed for the identified overlapping nodes. If not, it returns a list of identified overlapping nodes along with the overlapping community structure detected. This optional phase might result in over-detection in case of high overlapping density and diversity in the network. This occurs if the threshold set for identification of overlapping nodes based on neighbors to community ratio is not indicative of the network's characteristics.

The pseudocode presenting the proposed logic for post-processing is presented in Figure 4.6.

## 4.6 Optimizations

In this section, I discuss the optimizations that are carried out to make the implementation run faster. In addition, I provide a consolidated list of the algorithm parameters. During seed selection, a new seed is selected only if it does not belong to any of the communities constructed earlier. To achieve this, a naive implementation would look at every node in the communities constructed so far before it selects a new seed. IDLE improves this operation by maintaining a dynamic $O(|V|)$ buffer that keeps track of all the nodes that have been included earlier during local expansions. This improves new seed selection into an $O(1)$ operation. For the local expansion step, I adopt optimizations from GCE [43]. IDLE dynamically updates the 1-neighborhood and internal and

**Figure 4.6** Algorithm POST-PROCESSING

**Input: G(V, E, W)**, the graph representtion of the complex network,
**Communities**, a map structure that contains the list of communities for each node in G,
**Output: OverlappingNodes**, a list containing the overlapping nodes in G

```
 1: $OverlappingNodes \leftarrow [\,]$
 2: for each node $v \in V$ do
 3:     $N \leftarrow$ G.NEIGHBORS($v$)
 4:     $T \leftarrow \{\}$
 5:     for each neighbor $v \in N$ do
 6:         for each community $c \in Communities[n]$ do
 7:             $T[c] \leftarrow T[c] + 1$
 8:         end for
 9:     end for
10:     $flag \leftarrow 0$
11:     for each community $c \in T$ do
12:         if $\frac{T[c]}{\text{SIZE}(N)}$ is detected to be the majority proportion then
13:             $flag \leftarrow 1$    ▷ Majority of the node's neighbors belong to a single community, so it
                cannot be overlapping
14:         end if
15:     end for
16:     if flag == 0 then
17:         $OverlappingNodes$.ADD($v$)
18:     end if
19: end for
20: return $OverlappingNodes$
```

external degrees of the entities involved, resulting in faster fitness calculations. These operations are shown in Figure 4.4. GCE is shown to be faster than other similar local expansion techniques in [43]. In fact, many operations performed in GCE are unnecessary in IDLE because of our improved seed selection process. Hence IDLE is guaranteed to be faster than many previously known overlapping community detection techniques. The parameters defined for IDLE are discussed is Table 4.2.

Table 4.2: Table showing the parameters defined for IDLE

| **Parameter** | **Description** | **Default values** |
|---|---|---|
| $p$ | Threshold for the proportion of neighbors of a seed node that are already assigned to communities. If observed proportion $> p$, corresponding node is an overlapping node. | 0.5, if degree of node $\leq 4$ and 0.8, otherwise. |
| $w$ | Threshold for the value of community degree ratio (CDR) of a node $v$ in a community $c$. If CDR $\leq$ w for $v$, it is pruned from $c$. | 0.3 or 0.4. Usually decided based on average fitness of communities detected. |
| $\epsilon$ | Threshold for the overlap coefficient between two communities $c_i$ and $c_j$, used in duplicate detection. If overlap coefficient($c_i$, $c_j$) $\geq \epsilon$, a duplicate community is detected. | 0.6 |
| $\gamma$ | An absolute value of count or fraction of neighbors of a node that belong to a community, before the node itself can be assigned to that community. Used in assigning communities for detected overlapping nodes. | Default value used is an absolute count of 1 or 2 based on node's connectivity. |

It is important to note that the default values for these parameters were determined from experiments on networks with different attributes. Though these parameters were learned from synthetic networks, we have used the same default values for experiments on real networks whose properties are not known in advance. From Chapter 5, we can see that IDLE performs very well on real networks, measured in terms of the extended modularity and extended modularity density values of the covers. This confirms that the default parameters are in fact appropriate for complex networks in general and there is no overfitting for a certain type of networks. However, these are designed to be flexible and one can explore the perrformance of IDLE using different values for the parameters and determine the best suited values for a given network. The pseudocode for IDLE is presented in Figure 4.7.

**Figure 4.7** Algorithm IDLE

**Input: G(V, E, W)**, weighted/unweighted graph representation of the complex network
**Output: Cover**, a list of list structure containing list of detected overlapping communities, **OverlappingNodes**, a list of detected overlapping communities

```
1: SORT-BY-DEGREE($V$, '$descending$')
2: $Partition \leftarrow$ K-CORE-DECOMPOSITION($G$)    ▷ Partition contains $k$-core values for each node
3: $k \leftarrow$ GET-MAX-K($Partition$)
4: $Covered \leftarrow \{\}$
5: $C \leftarrow [\ ]$
6: $OverlappingNodes \leftarrow [\ ]$
7: $j \leftarrow 0$
8: while #NodesCovered $< |V|$ do
9:     $CurrentCore \leftarrow$ GET-K-CORE($k$)    ▷ Get the list of nodes in $k$thcore
10:    for each node $i \in CurrentCore$ do
11:        if $i \notin Covered$ then    ▷ i is selected as a seed
12:            $C_j \leftarrow$ LOCAL-EXPANSION($G, i, Covered, Cover$)
13:            if $c_j \neq [\ ]$ then
14:                $\#NodesCovered \leftarrow \#NodesCovered + |C_j|$
15:                $Covered \leftarrow Covered \cup C_j$
16:                $j \leftarrow j + 1$
17:                $Cover$.ADD($C_j$)
18:            else
19:                $OverlappingNodes$.ADD($i$)
20:            end if
21:        end if
22:    end for
23:    $k \leftarrow k - 1$
24: end while
25:
26: ASSIGN-COMMUNITIES($OverlappingNodes, \gamma$)    ▷ Assign communities to overlapping nodes
    identified during local expansion, using $\gamma$ or default count values
27:
28: $OverlappingNodes \leftarrow$ POST-PROCESSING($G, Cover$)
29: return $Cover, OverlappingNodes$
```

# Chapter 5

# Experiments and Results

In this chapter, I discuss the different experiments that were conducted to analyze the performance of IDLE, which is a local expansion based technique for detecting overlapping community structures of complex networks. In Chapter 3, I have discussed other overlapping community detection techniques that are considered state-of-the-art. To compare the performance of IDLE with the state-of-the-art, I have chosen two of the best performing algorithms shown in the comparative study conducted by Xie et al. [73]. One of them is SLPA [75], the speaker-listener interaction based label propagation algorithm, and the other one is GCE [43], the greedy clique expansion technique for detecting highly overlapping communities. I have selected SLPA for this comparative study because Xie et al. [73] conclude that it is the top-ranked algorithm for overlapping community detection in both real and synthetic networks, and I believe it is meaningful to evaluate the performance of IDLE against this technique. The reason for selecting GCE is that it is a local expansion based community detection technique that outperforms all the other local expansion techniques (as observed from the results presented in [73]). Since IDLE also falls in the same category, it is useful to compare its performance against that of GCE.

In my experiments, I have used both synthetic and real networks to study the performance of IDLE, SLPA, and GCE. All three algorithms produce crisp assignments of nodes to communities. Xie et al. [73] present extensive analyses of results of the 14 algorithms they considered. This helps the reader get a clear idea of their performances from all aspects of their ability to detect ground-truth communities, from the measure of closeness of detected to ground-truth communities to the distribution of detected community sizes. In this chapter, I have adopted a similar style to present the results from my experiments. I first discuss the tests conducted on synthetic networks (Section 5.1) followed by the tests on real networks (Section 5.2).

## 5.1 Tests on Synthetic Networks

Though there are many real-world complex networks available today, more often than not, one does not know the ground-truth communities present in them. This makes the testing of community detection algorithms difficult. The solution to this important problem is to build synthetic networks that resemble real networks in their features and whose ground-truth community structure is known.

Synthetic networks are computer-generated graphs, with certain properties, that are created using specialized generation algorithms. There are many graph generation algorithms proposed [40, 61, 54, 42], and they come with parameters whose values can be controlled to generate benchmark networks with desired properties.

### 5.1.1 LFR Benchmarks

The earliest set of benchmark graphs for community detection were proposed by Girvan and Newman [54] to test their edge betweenness-based disjoint community detection algorithm. However, this benchmark has limitations with respect to heterogeneity in node degree distribution and community sizes, and more importantly, it does not account for overlapping communities in networks. These limitations render this benchmark unusable in our context. The LFR benchmarks is a class of benchmark graphs proposed by Lancichinetti et al. [40] to generate synthetic networks that closely resemble real ones. The LFR benchmark is widely used for experiments in overlapping community detection due to the rich set of parameters it provides for graph generation. By setting these parameters to appropriate values, one can generate benchmarks with properties that highlight unique strengths as well as limitations of community detection algorithms.

The synthetic networks used in experiments discussed in this section belong to the class of LFR benchmark graphs. To understand properties of these networks, it is important to be aware of the parameters provided by the LFR model:

- $N$ is the number of nodes in the network.
- $\tau_1$ and $\tau_2$ are the exponents for power law distributions of node degree and community size repectively.
- $\langle k \rangle$ is the average degree of nodes, and $k_{max}$ is the maximum degree.
- $\mu$ is the mixing parameter. That is, every node shares a fraction $1 - \mu$ of its edges with nodes in its community and a fraction $\mu$ of its edges with nodes in the remainder of the network.
- $s_{min}$ and $s_{max}$ denote the minimal and maximal sizes of communities in the graph.
- $O_n$ is the desired number of overlapping nodes in the network.
- $O_m$ is the number of memberships for overlapping nodes (i.e., the number of communities the overlapping nodes simultaneously belong to). The non-overlapping nodes naturally belong to a single community.

The only drawback of this benchmark is that it assigns all overlapping nodes to the same number of communities, which is not true in real networks. However, the LFR benchmark produces networks that have other properties very close to those observed in real networks and is hence widely used to test overlapping community detection algorithms.

### 5.1.2 Normalized Mutual Information

Normalized mutual information (NMI) is a measure borrowed from information theory that was initially adopted to compare two partitions [22]. It was one of the preferred measures for computing similarity of paritions due to its sensitivity to errors in community detection. Lancichinetti et al. [41] propose an extension to NMI that can be used to compare two covers $C'$ and $C''$. Therefore, given the ground-truth cover of the network and the cover detected by an overlapping community detection algorithm, NMI helps us measure how close the detected communities are to ground-truth, thereby providing an evaluation of the ability of the algorithm to detect the right communities.

As we know, a node in a cover may belong to more than one community. Let $x_v$ denote the membership of node $v$ in cover $C'$. The node membership in case of a cover can be expressed as an array of size $|C'|$ with binary entries representing if the node belongs to the corresponding community (i.e., $(x_v)_k = 1$ if node $v$ belongs to community $c'_k$, $(x_v)_k = 0$ otherwise). The $k$th entry of this membership array can be regarded as a realization of a random variable $X_k$, that corresponds to community $c'_k$ in cover $C'$, whose probability distribution is given as

$$P(X_k = 1) = \frac{n_k}{N}, P(X_k = 0) = 1 - \frac{n_k}{N}.$$

where, $n_k$ is the size of community $c'_k$. $N$ is the total number of nodes in the network. Similarly, we can define a random variable $Y_l$ assiciated with community $c''_l$ in cover $C''$. Their joint probability distribution $P(X_k, Y_l)$ is defined based on the covers $C'$ and $C''$, and this is used in entropy calculation.

$H(X_k)$ is the entropy of the random variable $X_k$ associated with cover $C'$ and $H(Y_l)$ is the entropy of the random variable $Y_l$ associated with cover $C''$. In order to define how similar the covers are, *conditional entropy* $H(X_k|Y_l)$ is used to define the amount of information to infer $X_k$ given a community $Y_l$ in $C''$. That is

$$H(X_k|Y_l) = H(X_k, Y_l) - H(Y_l).$$

Since there are $|C''|$ possible candidates

$$H(X_k|Y) = \underset{l \in \{1,2,..,|C''|\}}{min} H(X_k|Y_l).$$

This is normalized as

$$H(X_k|Y)_{norm} = \frac{H(X_k|Y)}{H(X_k)}.$$

The normalized conditional entropy of $X$ with respect to $Y$ is defined as

$$H(X|Y)_{norm} = \frac{1}{|C'|} \sum_k \frac{H(X_k|Y)}{H(X_k)}.$$

$H(Y|X)_{norm}$ can be defined similarly. Using these, normalized mutual information is defined as

$$NMI(X|Y) = 1 - \frac{1}{2} \left[ H(X|Y)_{norm} + H(Y|X)_{norm} \right].$$

$NMI$ for two covers $C'$ and $C''$ is in the range $[0, 1]$, and it is equal to 1 if and only if the two covers are equal. I use NMI as one of the measures to evaluate the quality of covers detected by IDLE, SLPA and GCE in the experiments.

### 5.1.3 Omega Index

Hubert and Arabie [30] proposed the adjusted Rand Index to measure the similarity between two partitions based on the degree to which pairs of nodes are placed in agreement in communities. In partitions, each node occurs in only one cluster and hence their similarity can be measured in terms of the number of pairs of nodes that are placed together or apart in both the partitions. But in covers, nodes can belong to more than one community and hence it is possible for pairs of nodes to be placed together more than once. Collins and Dent [21] proposed an extension of the adjusted Rand Index called the *Omega Index* that quantifies similarity between covers by measuring the number of pairs of nodes that are in agreement in the same number of communities. Conceptually, Collins and Dent express the adjusted Rand Index (that the Omega Index is an extension of) as

$$\text{final index} = \frac{\text{observed index - expected index}}{\text{maximum index - expected index}}.$$

This is explained in [21] as the observed improvement over chance divided by the maximum improvement possible over chance. The maximum index always takes the value 1.

In Omega Index, the observed index $\omega_u(C', C'')$ is a simple extension of the unadjusted Rand Index to account for covers and it is the fraction of pairs of nodes that appear together in the same number of communities. It is given as

$$\omega_u(C', C'') = \frac{1}{N} \sum_{j=0}^{min(K_1, K_2)} |t_j(C') \cap t_j(C'')|.$$

$N$ is the number of pairs of nodes and it equals $n(n-1)/2$ if the number of nodes in the network is given as n. $K_1$ and $K_2$ are the number of communities in covers $C'$ and $C''$ respectively. $t_j(C)$ is the set of pairs of nodes that appear together in exactly j communities in cover C.

The expected index $\omega_e(C', C'')$ is the expected value of the fraction in a null model, and it is

$$\omega_e(C', C'') = \frac{1}{N^2} \sum_{j=0}^{min(K_1, K_2)} |t_j(C')| \cdot |t_j(C'')|.$$

Substituting these values in the conceptual notation of omega index, Gregory [28] expresses the omega index of two covers $C'$ and $C''$ as

$$\omega(C', C'') = \frac{\omega_u(C', C'') - \omega_e(C', C'')}{1 - \omega_e(C', C'')}.$$

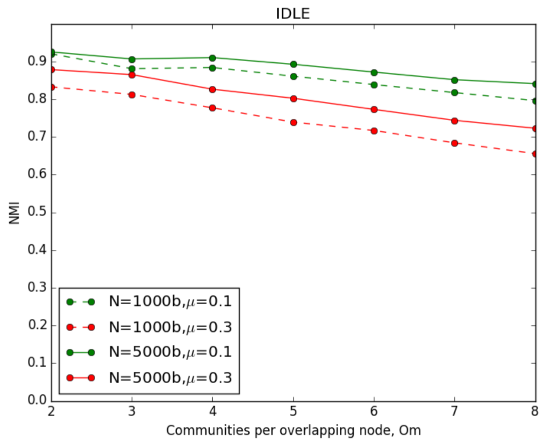
If there is no overlap between communities in both the covers, the omega index reduces to adjusted Rand Index. The omega index takes a value 1 if there is a perfect match between the two covers. In addition to NMI, I have also used Omega Index to show the performance of IDLE and the other algorithms.

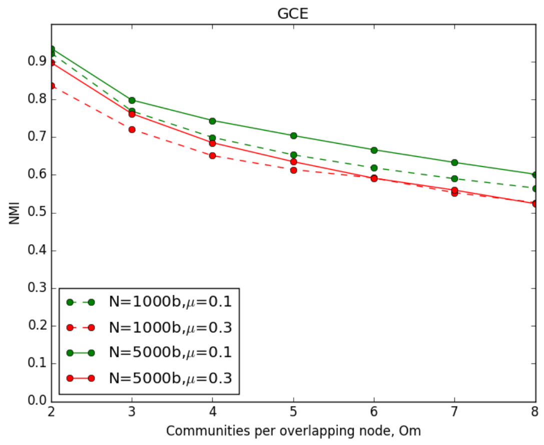
### 5.1.4 Effects of Benchmark Parameters on Quality of Detection

In this section, I present how the quality of detection (measured in NMI) is affected by different parameters of the LFR benchmark. For this, I have generated synthetic networks with number of nodes $N$ from the set $\{1000, 5000\}$. The average degree of nodes $\langle k \rangle$ is set to 10 and Xie et al. [73] observe that this is of the same order as many large real-world social networks presented in [47]. The other parameters are: power law exponents $\tau_1 = 2$ and $\tau_2 = 1$, and maximum degree $k_{max} = 50$. The mixing parameter $\mu$ is set to one of 0.1 or 0.3 for different experiments. Two ranges are selected for community sizes, $s = [10, 50]$ and $b = [20, 100]$. The value for the number of overlapping nodes $O_n$ is set to either 10% or 50% of the total number of nodes. The number of memberships $O_m$ for overlapping nodes is varied from 2 to 8, making the detection of overlapping communities more difficult at each level. By setting benchmark parameters to be the same as those used in [40] and [73] I believe that the reader can obtain a good idea of how IDLE performs not only against the algorithms discussed in this thesis but also other overlapping community detection techniques discussed in comparative studies in literature.

Each of the algorithms used in our study comes with its own set of parameters. It is important to discuss the values set for these parameters to be able to reproduce results shown in this thesis. For IDLE, the value for threshold $\epsilon$, which determines if a candidate community is a duplicate, is set to 0.75. I disable phase 2 detection of overlapping nodes in all the experiments since the $\gamma$ value for these networks is not known in advance. Inspite of this, IDLE detects a good fraction of the overlapping nodes when compared to GCE and SLPA, as shown in Section 5.1.6. For GCE, we set all parameters to their recommended default values as used by Lee et al. [43]. The values are, $k = 4$, $\alpha = 1.0$ and $\epsilon = 0.6$. Since SLPA is non-deterministic, we run the algorithm 10 times on each instance of the network with values for $r$ in the range $[0.05, 0.5]$ with a difference of 0.05 between consecutive values. For a network instance, the best value obtained for NMI from these multiple runs, which corresponds to any $r$ value in the given range, is reported. In practical scenarios, one does not usually know the best value for $r$ to be used for a network. So I believe that the average value of NMI obtained for a network instance better reflects the performance of SLPA. Keeping this in mind, I present comparative study plots for both the best and average values of NMI obtained for the networks in Figures 5.3, 5.4, 5.5 and 5.6. To obtain each point in the plots shown in all the figures in this section, the algorithms were run on the same 10 instances of graphs generated with parameters chosen for that experiment. The average of these values is reported.

Figure 5.1 shows the performance of the three algorithms for different network sizes and mixing parameters. The solid lines correspond to results from networks with $N = 5000$ and the dotted lines represent results from networks with $N = 1000$ nodes. The colors represent mixing parameter values: green curves correspond to $\mu = 0.1$ and red curves to $\mu = 0.3$. From these plots, we observe that the mixing parameter has a greater effect on the performance of algorithms as this determines the ratio of edges a node shares with other nodes outside its community. The higher this value, the less tight a community becomes, making its detection difficult. Also, the algorithms perform better for larger networks ($N = 5000$). On the $x$-axis, I have increasing values of $O_m$, the number of memberships for overlapping nodes. As this value increases, community detection becomes difficult, and this is reflected in the deteriorating performance of the algorithms. The communities are large with their size in the range $b = [20, 100]$. From the plots for the three algorithms, we observe that the NMI values drop at a moderate rate for IDLE when compared to the other two.





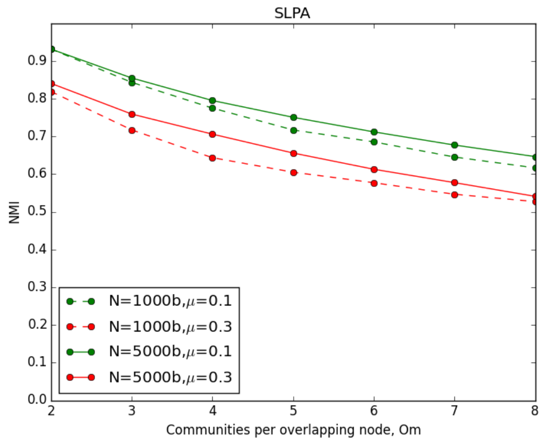


Figure 5.1: Plots showing the performance of IDLE, GCE and SLPA when applied to networks of different sizes and mixing parameters. It highlights the effect on performance caused by an increase in network size $N$, mixing parameter $\mu$ and overlapping node memberships $O_m$.

In Figure 5.2, I present the performances of IDLE, GCE and SLPA measured in NMI for different values of the number of overlapping nodes $O_n$ and community sizes in networks of size $N = 5000$. One can see that all three algorithms perform poorly when the number of overlapping nodes is set to 50% of the total number of nodes. This, when coupled with increasing values for number of memberships for the overlapping nodes, affects performance drastically. The performance of IDLE in the high overlap density case ($O_n = 50\%$) is not surprising because if more than 50% of the nodes in the network with average degree $\langle k \rangle = 10$ belong to $O_m > 2$ communities, one can infer that the communities in the network are not tightly defined. That is, there are many overlapping nodes in the network, and hence the number of external edges between communities is not sparse. This makes the detection task difficult, and almost all the algorithms studied in [73] show similar performance. The community size ranges used are, $s = [10, 50]$ and $b = [20, 100]$. As shown in the plots, the range of community sizes used does not cause much impact on the performances of these algorithms. The points shown for SLPA in both Figure 5.1 and 5.2 correspond to averages of the average NMI values obtained from multiple runs for each network instance.

In the following sections, I present the performances of IDLE, GCE and SLPA in the following:

- Detection of overlapping communities,
- Detection of overlapping nodes and
- Detection of overlapping node memberships

I measure the ability of an algorithm to detect overlapping communities by calculating NMI and Omega Index values of the detected communities using ground-truth. This gives one an idea of the ability of these algorithms to uncover overlapping communities planted in the benchmark networks used in experiments. The results from these experiments are shown in Section 5.1.5. In addition to this, it is also important to estimate the ability of an algorithm to detect overlapping nodes in the network along with the multiple communities they actually belong to. The ability of an algorithm to detect overlapping nodes is studied by tranforming it into a binary classification problem. This is described in detail along with the relevant results in Section 5.1.6. An algorithm succeeds in its task of detecting overlapping communities in a network when it can detect the multiple communities an overlapping node belongs to. The performance of our algorithms in detecting this is shown in Section 5.1.7.

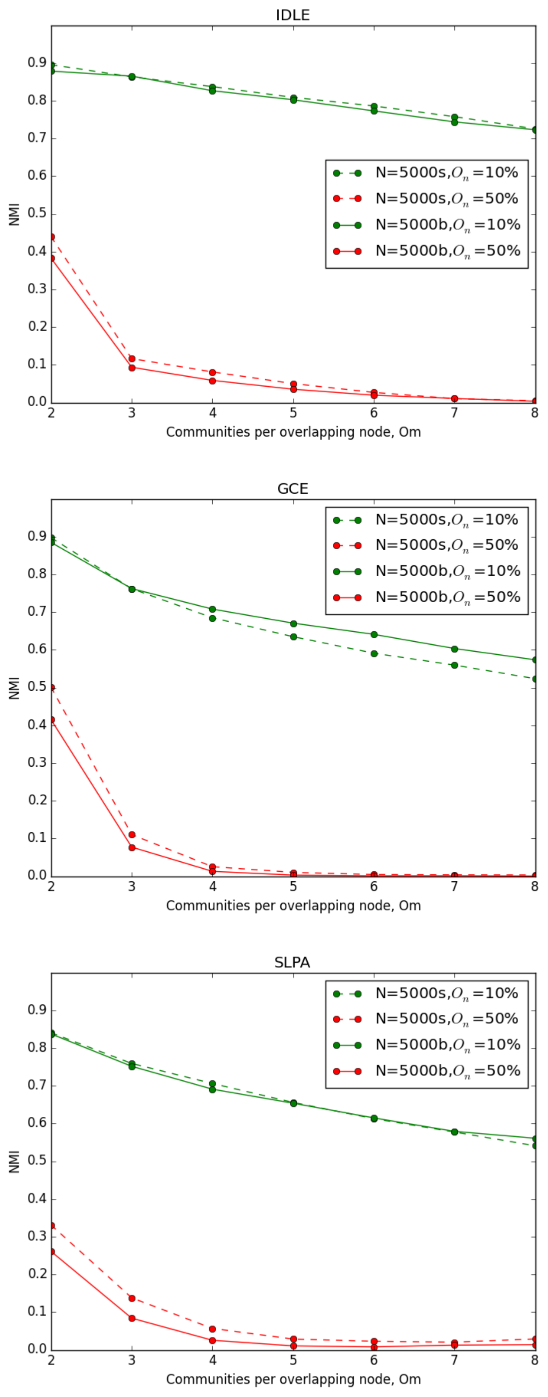
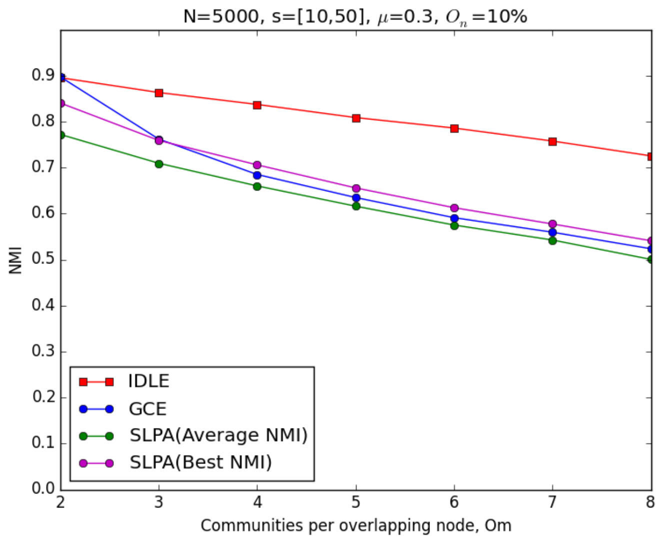


Figure 5.2: Plots showing the performance of IDLE, GCE and SLPA when applied to networks having different percentages of overlapping nodes and community sizes. It highlights the huge impact on performance caused by an increase in number of overlapping nodes $O_n$ along with their memberships $O_m$. The community size ranges are, $s = [10, 50]$ and $b = [20, 100]$.

### 5.1.5 Detection of Overlapping Communities

In this section, I present a comparative study of the ability of IDLE, GCE and SLPA to detect overlapping communities planted in benchmark networks. The results obtained are shown using both NMI and Omega Index in the following figures.



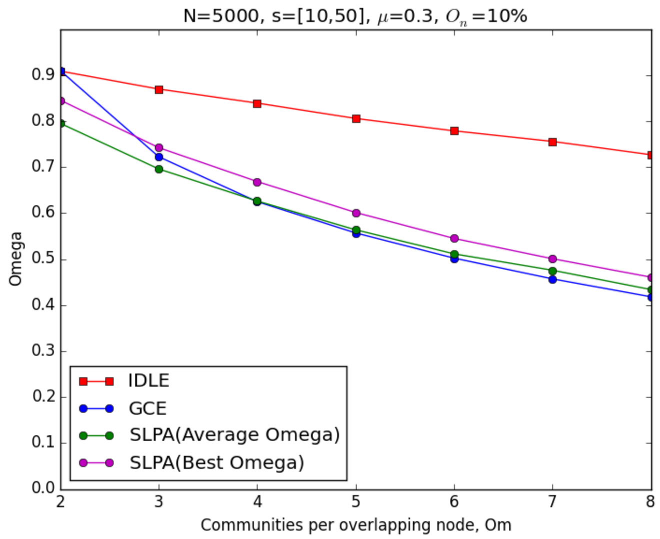
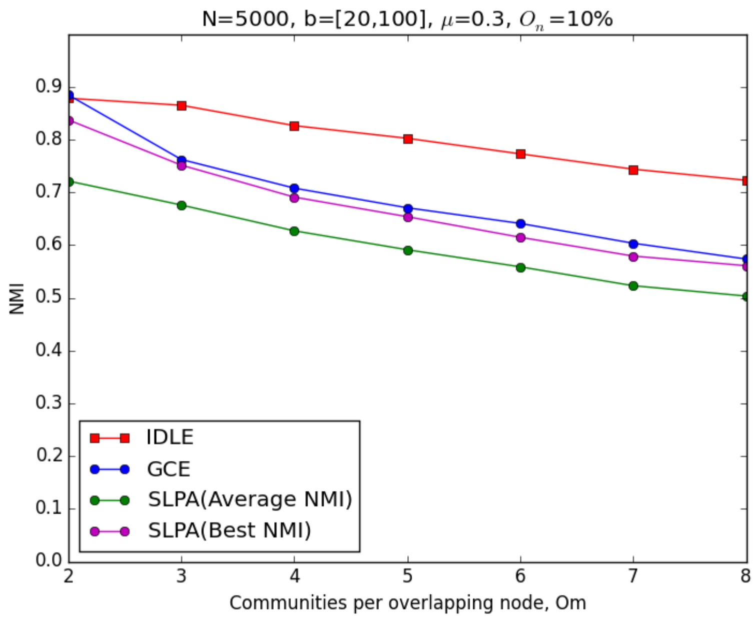


Figure 5.3: Comparative study plot showing the performances of IDLE, GCE and SLPA measured in NMI and Omega Index on networks with the following parameters: $N$=5000, $s = [10, 50]$, $\mu$=0.3, $O_n$=10%, for increasing values of overlapping node membership $O_m$.



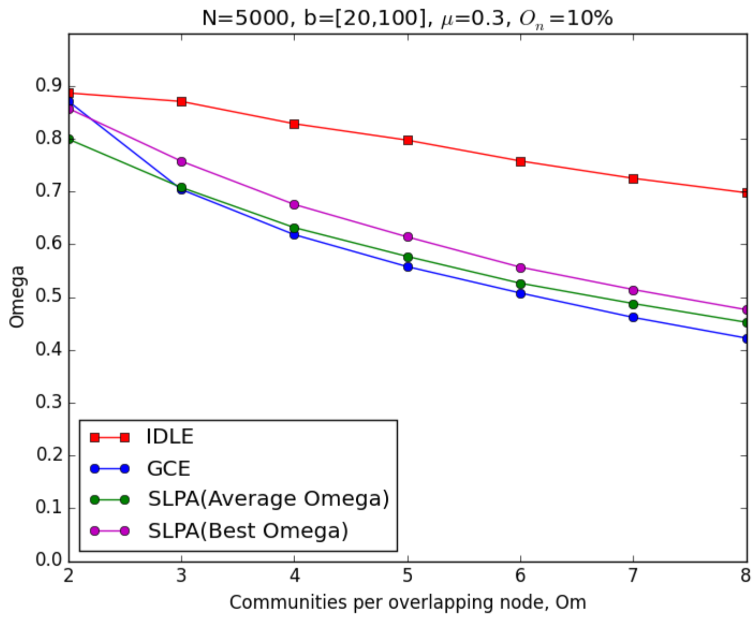
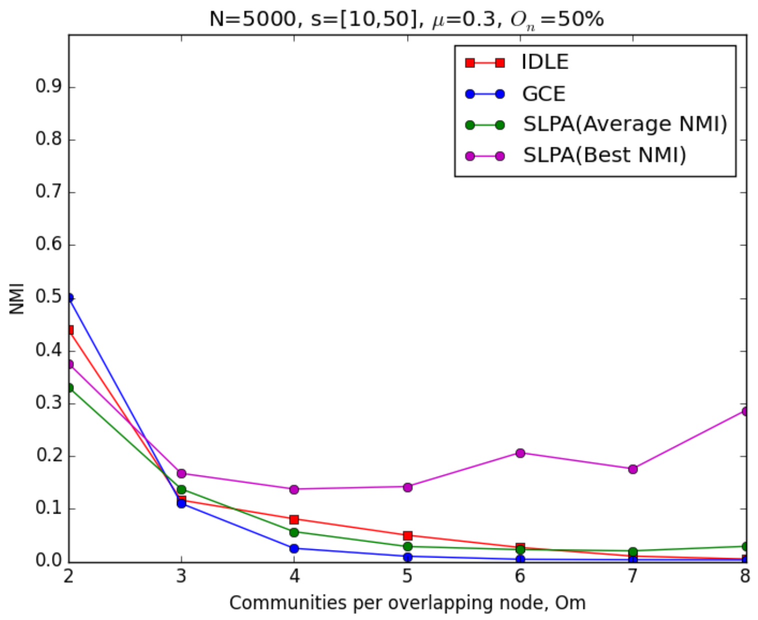


Figure 5.4: Comparative study plot showing the performances of IDLE, GCE and SLPA measured in NMI and Omega Index on networks with the following parameters: $N$=5000, $b = [20, 100]$, $\mu$=0.3, $O_n$=10%, for increasing values of overlapping node membership $O_m$.



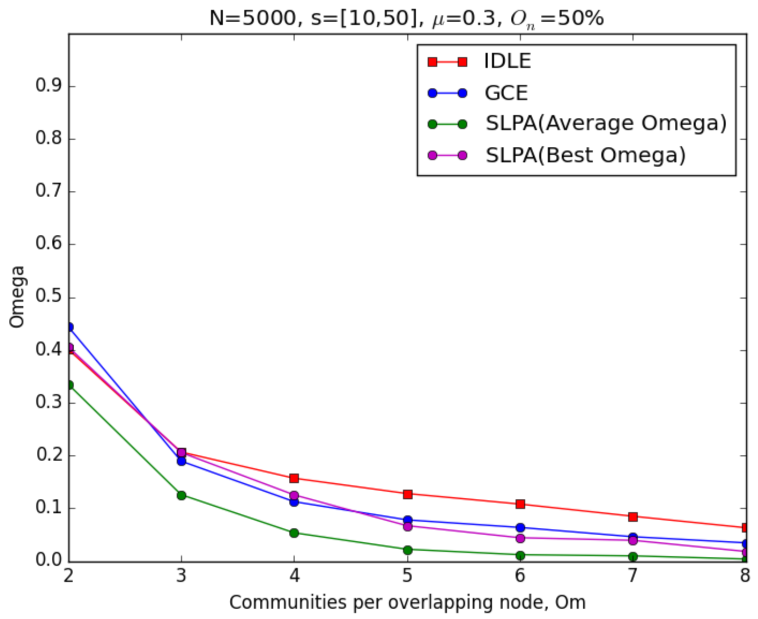
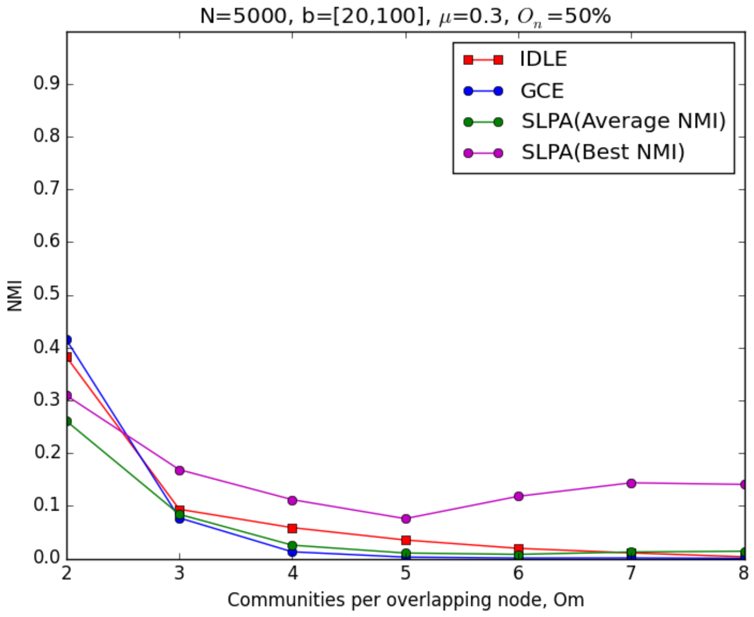


Figure 5.5: Comparative study plot showing the performances of IDLE, GCE and SLPA measured in NMI and Omega Index on networks with the following parameters: $N$=5000, $s = [10, 50]$, $\mu$=0.3, $O_n$=50%, for increasing values of overlapping node membership $O_m$.



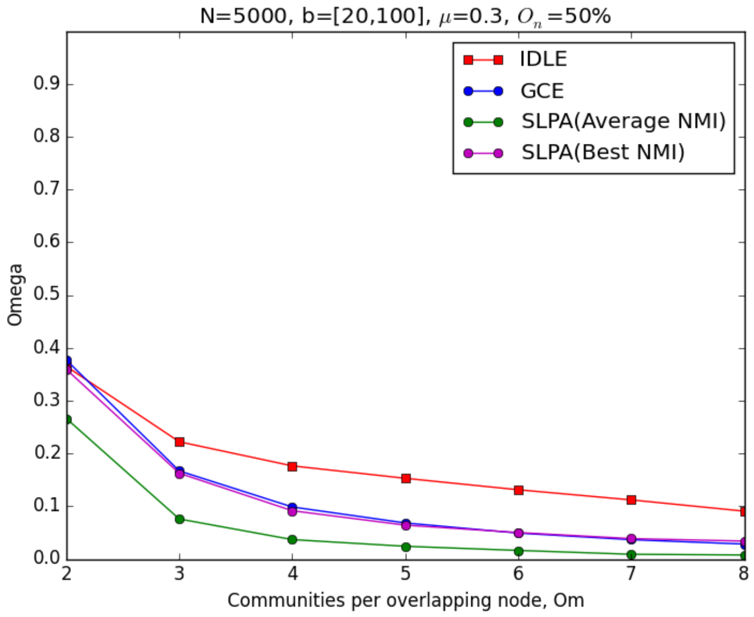


Figure 5.6: Comparative study plot showing the performances of IDLE, GCE and SLPA measured in NMI and Omega Index on networks with the following parameters: $N$=5000, $b = [20, 100]$, $\mu$=0.3, $O_n$=50%, for increasing values of overlapping node membership $O_m$.

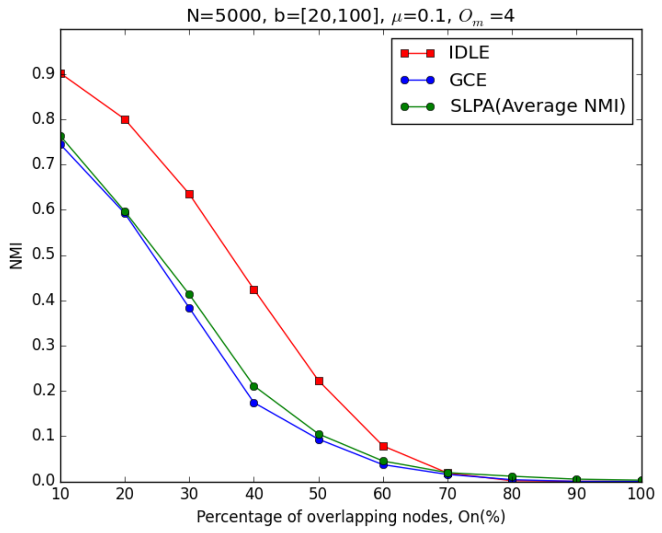
Each of the figures show performances of the algorithms on LFR networks generated with parameters shown above the corresponding plots. The results are shown using NMI and Omega Index values as a function of the number of memberships of overlapping nodes ($O_m$). As already discussed, there are benchmarks created in [43] and [73] and in earlier literature in this field to compare the performance of algorithms that are considered state-of-the-art. Since it is only logical to test IDLE on the same set of benchmarks, to get an idea of its performance in the bigger picture, I have replicated some of these experiments in my analysis. For SLPA, I have presented its performance using both the best and average NMI values obtained for each network instance.

One can observe that there is a positive correlation between the NMI and Omega index values presented in the plots; those algorithms that perform the best in terms of NMI also obtain the highest values for Omega index. This is violated in a few cases where the networks have high overlapping densities, as observed in Figures 5.5 and 5.6. From the results presented, we can clearly see that IDLE outperforms both SLPA and GCE in detecting overlapping communities, even when detection becomes harder (for increasing values of $O_m$) in networks where 10% of the nodes are overlapping. The range of community sizes ($s = [10, 50]$ and $b = [20, 100]$) do not affect this trend in performance (as shown in Figures 5.3 and5.4). There is a single exception to this and it is observed in networks having $O_m = 2$ as shown in the figures. GCE performs better than IDLE in this particular case, and this could be because of the way IDLE assigns communities to overlapping nodes detected from phase 1. When there is no $\gamma$ value specified, IDLE assigns an overlapping node to the communities of its neighbors that pass a default threshold. This might result in the assignment of overlapping nodes to more than 2 communites in some cases (though $O_m = 2$), resulting in a slightly lower performance. However, this is not observed in the Omega index values.

An interesting characteristic observed is that the performance of IDLE drops at a much slower rate for larger values of $O_m$ as compared to the other two algorithms. This is presented in both Figure 5.3 and 5.4. In fact, the NMI and Omega index values obtained for IDLE when $O_m = 8$ in these networks is better than those values obtained for GCE and SLPA for lower values of $O_m$ (say, $O_m = 4$). This improved performance of IDLE can be attributed to the way it approaches the problem of detecting overlapping communities, by not only identifying natural communities of seeds, but also focusing on explicitly detecting overlapping nodes in networks.

Figures 5.5 and 5.6 present the performances of IDLE, GCE and SLPA in detecting overlapping communities in networks with high overlapping density. In these benchmarks, 50% of the total number of nodes belong to multiple communities. As we can observe from the plots, all three algorithms perform poorly in this case. This is also the case with the 14 algorithms discussed in [73]. Though many of these algorithms perform well when the networks have lower overlapping nodes, their performances deteriorate drastically as this number increases. This brings out the fact that the algorithms we currently have are not equipped to detect overlapping communities in networks with high overlapping density (high values of $O_n$ and $O_n$). To demonstrate the effect of increasing values of overlapping node count ($O_n$) in networks in the case of a moderate number of memberships for overlapping nodes, I have conducted another experiment using LFR benchmark networks. The parameters for this experiment are: network size, $N = 5000$ nodes, average degree $\langle k \rangle = 10$, maximum degree $k_{max} = 50$, community size range $b = [20, 100]$, mixing parameter $\mu = 0.1$ and number of memberships for overlapping nodes $O_m = 4$. The value for number of overlapping nodes $O_n$ is increased from 10% to 100% of the total number of nodes in the network.

The performances of IDLE, GCE and SLPA in terms of NMI and Omega Index are measured and shown in Figure 5.7. We can see that performance drops drastically as more nodes become overlapping, and this is observed in the performances of all three algorithms. However, IDLE clearly has better NMI values than the other two as long as the number of overlapping nodes is less than 70%. In the case of Omega index, it consistently outperforms GCE and SLPA.



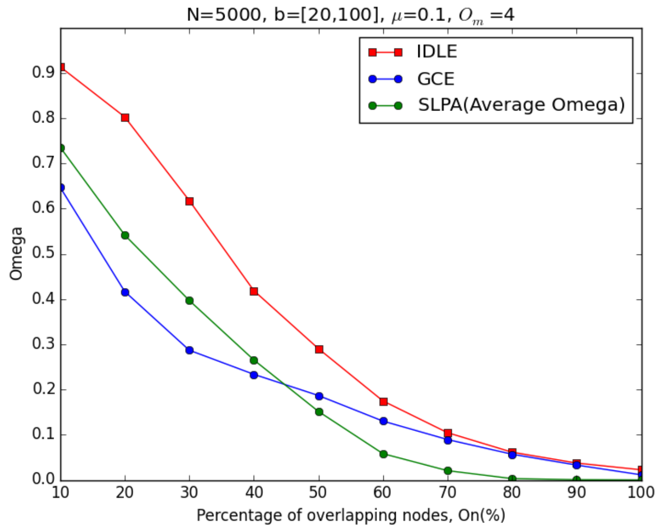


Figure 5.7: Comparative study plot showing the performances of IDLE, GCE and SLPA measured in NMI and Omega Index on networks with the following parameters: $N$=5000, $b = [20, 100]$, $\mu$=0.3, $O_m$=4, for increasing percentage of overlapping nodes $O_n$.

### 5.1.6 Detection of Overlapping Nodes

An important aspect of overlapping community detection algorithms that is not considered for disjoint community detection is their ability to detect overlapping nodes in the network. When an algorithm returns the communities it has detected on a network, there may be nodes that belong to more than one community. These nodes are labelled as overlapping nodes, and in this section I present the performance of IDLE, GCE and SLPA in detecting overlapping nodes in networks. Given two covers of a network, $C'$ and $C''$, NMI quantifies the performance of an algorithm in detecting overlapping communities from an information theory stand-point, by measuring the amount of information required to infer communities in one cover, say $C'$ given the other cover, say $C''$. Omega Index measures the number of pairs of nodes that are in agreement in the same number of communities in covers $C'$ and $C''$ and thereby quantifies the ability of an algorithm to retrieve overlapping communities in a network. But both NMI and Omega Index do not give a clear idea of the peformance of community detection algorithms in detecting overlapping nodes.

To quantify the performance of algorithms in detecting overlapping nodes, Ball at al. [8] use Jaccard Similarity Index defined as

$$J = \frac{|S \cup V|}{|S \cap V|}.$$

$S$ is the set of actual overlapping nodes in the network and $V$ is the set of overlapping nodes as identified by the algorithm. In [73], Xie et al. formulate the identification of overlapping nodes as a binary classification problem where the nodes that an algorithm identifies as "overlapping" are given label 1 and the non-overlapping nodes take label 0. Similarly, the actual labels for nodes are determined from the ground-truth community structure of the network. Using these labels, the $F1$ score of this classification is calculated as

$$F1 = 2 * \frac{precision * recall}{precision + recall}.$$

*precision* measures the ability of an algorithm to not misidentify a non-overlapping node as "overlapping", and it is defined in our context as

$$precision = \frac{Number\ of\ nodes\ correctly\ identified\ as\ overlapping}{Total\ number\ of\ nodes\ identified\ as\ overlapping}.$$

Hence, precision measures the quality of detected overlapping nodes. *recall* measures the ability of an algorithm to detect all overlapping nodes in the network, and it is defined in our context as

$$recall = \frac{Number\ of\ nodes\ correctly\ identified\ as\ overlapping}{Total\ number\ of\ overlapping\ nodes\ in\ the\ network}.$$

Recall quantifies the detected overlapping nodes in comparison to the actual overlapping nodes present. Both precision and recall contribute equally to the F1 score calculated. Since this measure captures the performance of an algorithm in detecting overlapping nodes both in terms of quality and quantity, we adopt the F1 score to present the performances of IDLE, GCE and SLPA in Figures 5.8, 5.9, 5.10 and 5.11. In these figures, we also explicitly present the contributions of precision and recall to the corresponding F1 score. All these quantities are expressed as a function of the number of memberships for overlapping nodes expressed as $O_m$.

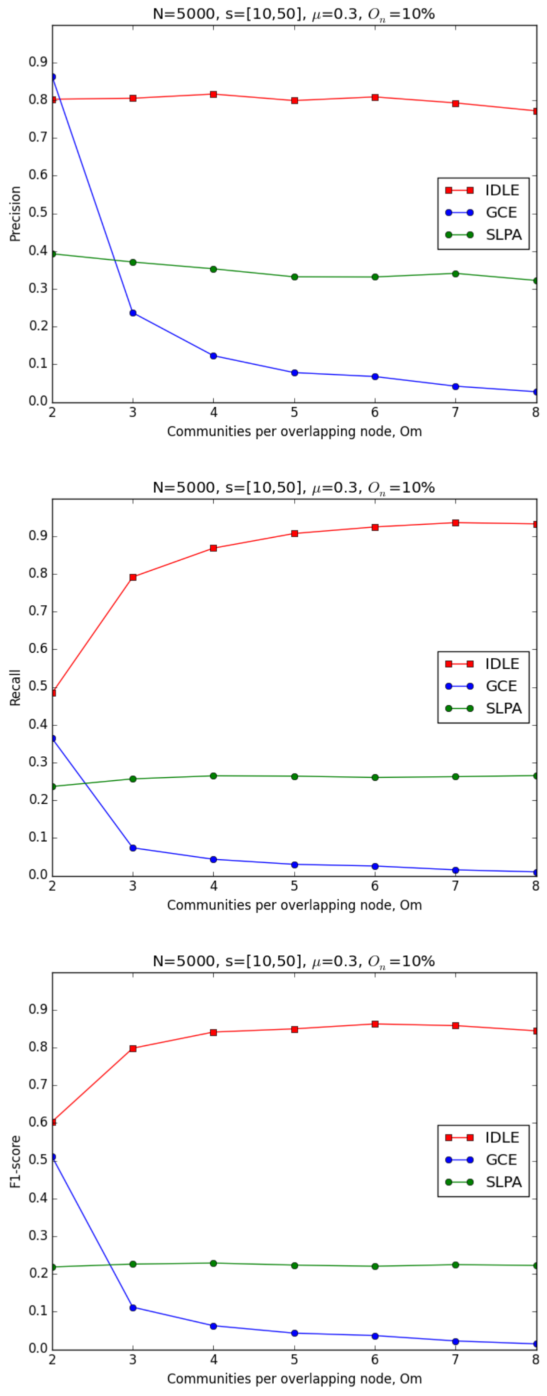


Figure 5.8: Comparative study plot showing the performances of IDLE, GCE and SLPA in detecting overlapping nodes in networks with the following parameters: $N$=5000, $s = [10, 50]$, $\mu$=0.3, $O_n$=10%, for increasing values of overlapping node membership $O_m$.

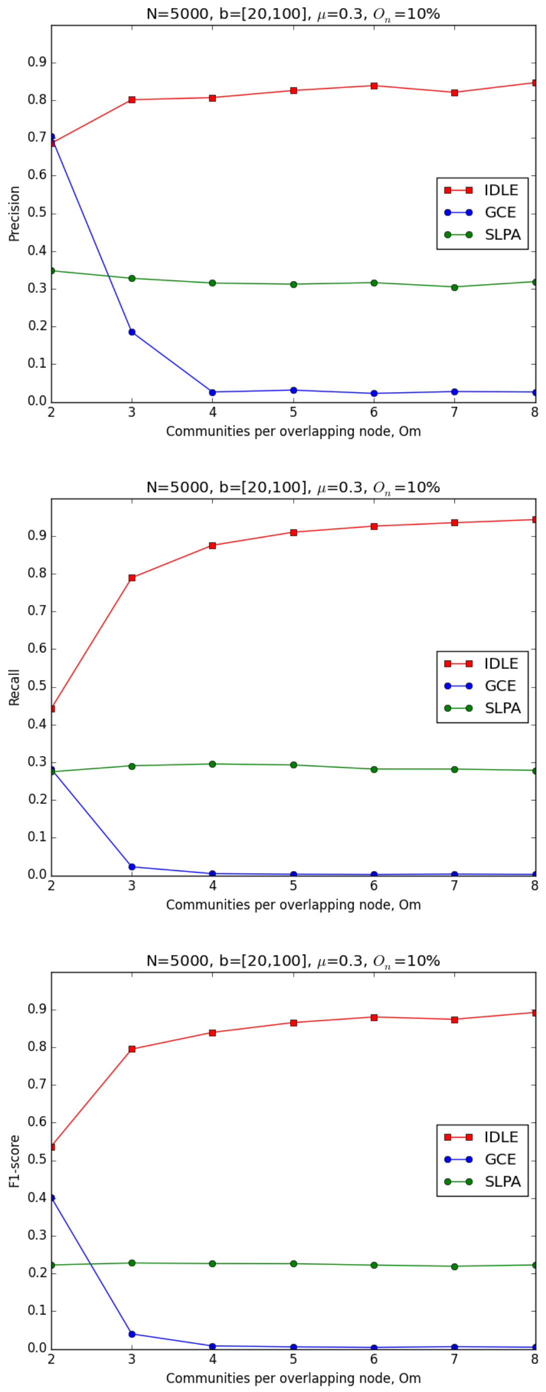


Figure 5.9: Comparative study plot showing the performances of IDLE, GCE and SLPA in detecting overlapping nodes in networks with the following parameters: $N$=5000, $b$ = $[20, 100]$, $\mu$=0.3, $O_n$=10%, for increasing values of overlapping node membership $O_m$.

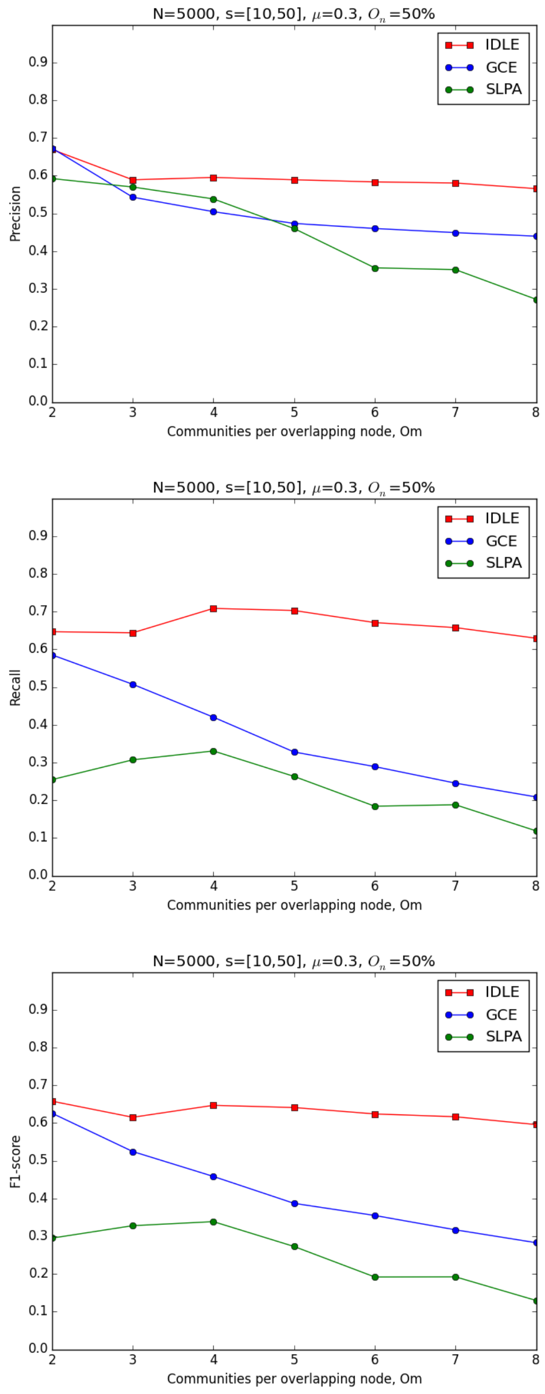


Figure 5.10: Comparative study plot showing the performances of IDLE, GCE and SLPA in detecting overlapping nodes in networks with the following parameters: $N$=5000, $s = [10, 50]$, $\mu$=0.3, $O_n$=50%, for increasing values of overlapping node membership $O_m$.

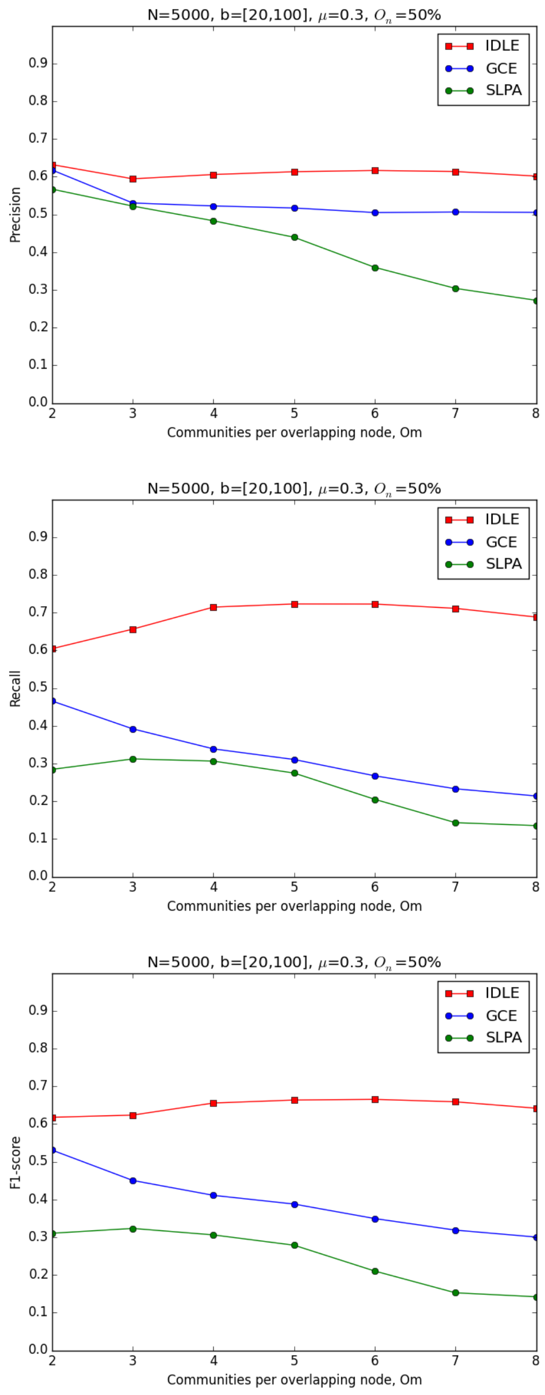


Figure 5.11: Comparative study plot showing the performances of IDLE, GCE and SLPA in detecting overlapping nodes in networks with the following parameters: $N$=5000, $b = [20, 100]$, $\mu$=0.3, $O_n$=50%, for increasing values of overlapping node membership $O_m$.

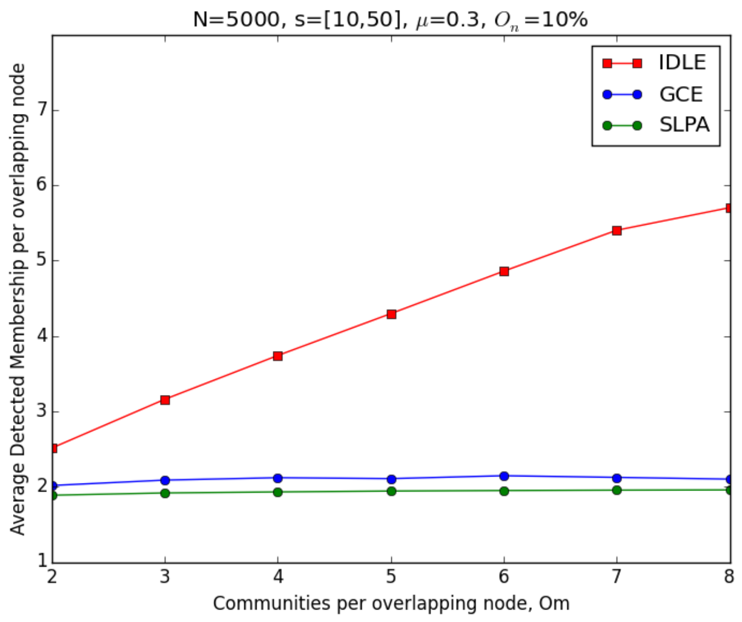
From the results presented in Figures 5.8, 5.9, 5.10 and 5.11, it is clear that IDLE outperforms the other algorithms in detecting overlapping nodes in the network. This is observed even in the case of networks with high overlapping density (as shown in Figures 5.10 and 5.11). To understand the contributions of precision and recall to the F1 score, I have also presented their values as a function of $O_m$. Both precision and recall increase with the value of $O_m$, and this is not observed in either GCE or SLPA. Their performances drop as the overlapping nodes belong to more communities.

Some algorithms like EAGLE [64] and Link [2] studied by Xie et al. [73] show an imbalance in precision and recall values. That is, they exhibit high precision combined with a low recall value or vice versa. This imbalance in turn contributes to underdetection or overdetection of overlapping nodes in a given network. This does not happen with IDLE, as there is no such imbalance observed in the precision and recall values.

### 5.1.7 Detection of Overlapping Memberships

In addition to detecting the overlapping nodes in a network, an algorithm should also be able to find all the communities the overlapping nodes belong to. This is an important factor that affects the quality of overlapping communities detected by an algorithm, especially as the nodes belong to more communities. In LFR benchmark networks, the membership count for overlapping nodes in ground-truth is given as $O_m$. The average number of memberships for overlapping nodes in the covers produced by these algorithms is computed for networks with increasing values of $O_m$. This is shown as a function of actual memberships in Figure 5.12. This shows the performance of IDLE, GCE and SLPA in detecting overlapping memberships of nodes networks. Each point in the plots is obtained by running the experiments on 10 instances of networks with the corresponding set of parameters. The average value obtained from the 10 instances is shown. For SLPA, the value taken for each network corresponds to the average of the detected memberships from multiple runs for different threshold($r$) values.

From the results shown in Figure 5.12, one can see that there is a clear difference between the performance of IDLE, GCE and SLPA in this aspect. Both GCE and SLPA always detect a constant (nearly 2) number of communities for overlapping nodes in networks with a few overlapping nodes (10% of total nodes), whereas IDLE exhibits a positive correlation with the number of memberships in ground-truth ($O_m$). A similar performance is also observed in the case where $O_n = 0.5$ of the total number of nodes for SLPA and IDLE. However, for GCE, the average number of detected memberships rises and then drops as $O_m$ becomes higher. I believe that this ability of IDLE to detect memberships of overlapping nodes better results in its improved performance. This is shown in the comparative study plots in Section 5.1.5, for increasing values of $O_m$, where the performances of GCE and SLPA drop at a faster rate due to their inability to detect multiple memberships of overlapping nodes correctly.



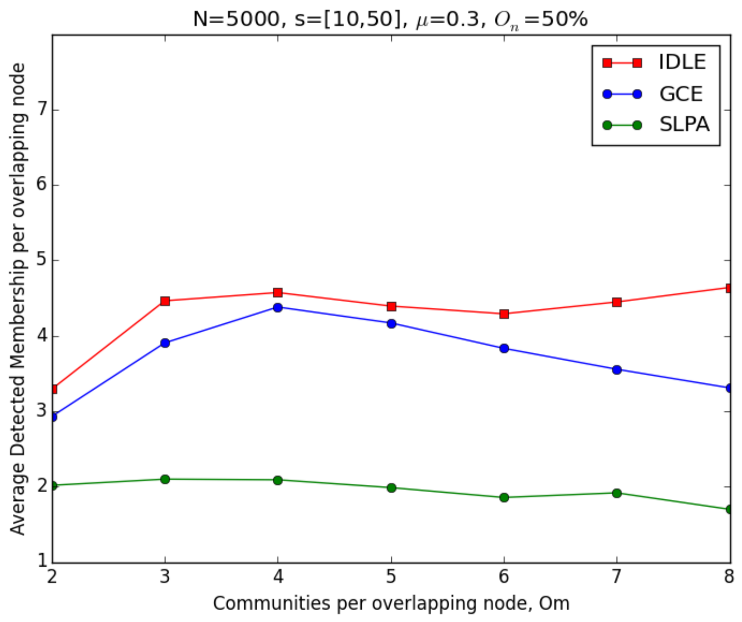


Figure 5.12: Comparative study plot showing the performances of IDLE, GCE and SLPA in detecting overlapping memberships on networks with the following parameters: $N$=5000, $s = [10, 50]$, $\mu$=0.3, $O_n$=10% and 50%, for increasing values of overlapping node memberships in ground-truth $O_m$.

### 5.1.8 Scalability

In this section, I discuss the scalability of IDLE on networks of increasing complexity. Since many complex networks today contain millions of nodes and billions of edges, it is important to understand how the performance of IDLE scales for such networks. This factor also influences the applicability of IDLE for real applications. In order to understand the scalability of IDLE, I have conducted two experiments similar to the ones conducted for GCE by Lee et al. [43]. In the first, IDLE is applied to LFR networks of increasing size ranging from $N = 10^2$ to $10^6$. The number of overlapping nodes also grow as a function of the network size and it is set as $O_n = 0.25$*N. The other parameters remain constant as the networks grow and are set as: $\langle k \rangle = 10$, $k_{max} = 50$, $\tau_1 = 2$, $\tau_2 = 1$, $c_{min} = 20$, $c_{max} = 100$, $\mu = 0.3$, and $O_m = 2$. The time taken by IDLE to produce covers for such networks is measured in *seconds* and the results are presented in Figure 5.13.

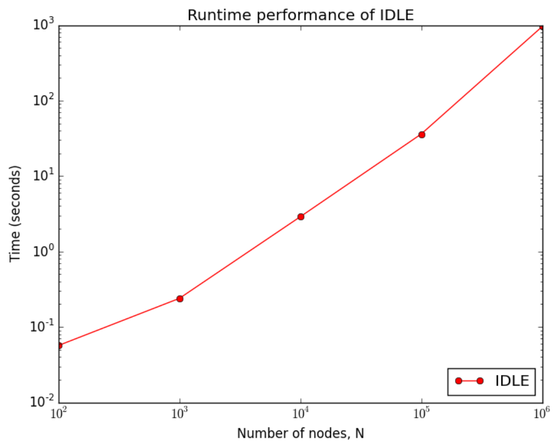


Figure 5.13: Time taken (in seconds) by IDLE to produce covers on networks of increasing size and proportional overlapping regions.

We observe that the runtime of IDLE scales similar to the runtime presented for GCE and other overlapping community detection algorithms. Since the implementation of these algorithms are provided in different languages, it is valuable to compare the trend in growth of runtime of these algorithms than to directly compare their runtime values on these networks.

The second experiment that is used to understand scalability of IDLE measures its performance on networks with growing average degree for nodes. In [43], Lee et al. note that this is one case where the performance of GCE degrades. The runtime of GCE increases rapidly as the average degree of nodes in the network increases. Since IDLE is also a local expansion based technique, I realize it is important to show its performance on a similar set of graphs. For this, I generate LFR networks using the same parameters shown in [43] for this experiment: $N = 5000$, the average degree $\langle k \rangle$ ranges from 20-180, $k_{max} = 200$, $\tau_1 = 2$, $\tau_2 = 1$, $c_{min} = \langle k \rangle$, $c_{max} = 500$, $\mu = 0.4$, $O_n = 0$, and $O_m = 1$. The time taken by IDLE to produce covers for these networks with increasing average degree is measured in *seconds* and the results are presented in Figure 5.14.

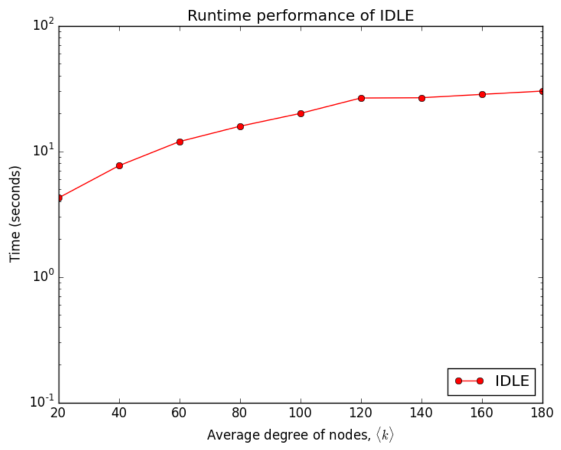


Figure 5.14: Time taken (in seconds) by IDLE to produce covers on networks of increasing average degree for nodes.

We observe that the runtime for IDLE on these networks does not show a rapid growth. This favorable scaling of IDLE results from the way it carries out seed selection. Unlike GCE, where all the maximal cliques of size above $k$ in the network are detected before local expansion, IDLE selects seeds based on their $k$-core value and performs a maximal clique listing only for the selected seeds. This, I believe, results in the difference observed in the performance of these algorithms on these networks.

## 5.2 Tests on Real Networks

Complex networks arise in many domains, as discussed in Chapter 1. Among their many properties, the division of nodes into groups called communities is commonly observed in networks due to the many interactions (represented as edges) that take place between these nodes [54]. However, we do not know the ground-truth community structures of these real networks. In some cases, though the community structure based on node attributes may be available, they might not be reflected in the actual network structure. Hence it is not easy to evaluate the performance of community detection algorithms on real complex networks.

To address this problem, Newman [55] proposed a quantity called *modularity* to measure the goodness of communities detected by algorithms designed for this purpose. This quantity is defined as

$$Q = \sum\nolimits_{c \in C} \left[ \frac{|E_c^{in}|}{|E|} - \left( \frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 \right].$$

Modularity Q is introduced and briefly discussed in Chapter 2. It is based on the fraction of edges between nodes in a community minus the fraction of edges that would connect the same set of nodes, if edges were placed between them randomly without regard to the community structure. Since this difference is computed for all the communities in a partition, modularity takes the form shown above. It can also be expressed as

$$Q = \frac{1}{2m} \sum\nolimits_{uv} \left[ A_{uv} - \frac{k_u k_v}{2m} \right] \delta_{c_u, c_v}.$$

Therefore, modularity quantifies how the connections between nodes in communities are better than random connections between them. A higher value for modularity corresponds to a better community structure. However, this formulation of modularity works well only in the case of disjoint community structures as it does not account for nodes that can belong to multiple communities. Hence it cannot be directly applied to measure quality of covers.

To make modularity suitable for overlapping community detection, various extensions are proposed and an overview of these measures is presented by Chen et al. [18]. In this section, I discuss the commonly used extensions of modularity, present the performances of IDLE, GCE and SLPA on real networks using these measures and show how these measures are still not good enough for evaluating performances of overlapping community algorithms on real networks.

### 5.2.1 Extended Modularity for Overlapping Community Detection

Chen et al. [18] present an overview of the different extensions of modularity proposed to account for overlapping nodes in the network. There are many node-based extensions of modularity [17, 51, 63, 64, 77], and an edge-based extension [56]. Since node-based extensions are more intuitive and their formulations are close to Newman's modularity, I have used them in this study. In disjoint community detection, each node belongs to a single community and the original modularity accounts for this scenario. However, in an overlapping community detection setup, nodes may belong to multiple communities at the same time. Given a cover $C = \{c_1, c_2, c_3, ..., c_k\}$, containing $k$ communities, where each $c_i$ corresponds to an individual community in the cover, extended modularities use the idea of *belonging coefficients.* The belonging coefficients quantify the strength of association of a node $v$ in the set of nodes $V$ to the communities in the cover $C$, and these are given as a vector $[\alpha_{vc_1}, \alpha_{vc_2}, \alpha_{vc_3}, ..., \alpha_{vc_k}]$. Each $\alpha_{vc_i}$ in the vector quantifies the strength of association of node $v$ to community $c_i$. The belonging coefficients are usually assumed to satisfy the following constraints:

$$0 \le \alpha_{vc} \le 1,$$
$$\sum_{c \in C} \alpha_{vc} = 1.$$

The different node-based extensions of modularity for overlapping community detection stem from different formulations for belonging coefficient values. There are two main notions based on which the belonging coefficients are defined and they correspond to the ideas of crisp and fuzzy assignment of nodes to communities. One assumes that a node belongs equally to all the communities it is assigned to and the other determines how tightly a node connects to a community to determine the strength of its association. In this section, I use extended modularity based on both these ideas of belonging coefficients to understand how they measure quality of communities in a cover.

Chen et al. [18] conclude that all node-based extensions of modularity can be expressed using

$$Q_{ov} = \sum\nolimits_{c \in C} \left[ \frac{|E_c^{in}|}{|E|} - \left( \frac{2|E_c^{in}| + |E_c^{out}|}{2|E|} \right)^2 \right].$$

Here, $|E_c^{in}|$, which represents the total number of internal edges in a community in Newman's modularity is given as, $|E_c^{in}| = \frac{1}{2} \sum_{u,v \in c} f(\alpha_{uc}, \alpha_{vc}) A_{uv}$. $|E_c^{out}|$, that represents the total number of external edges from the nodes in a community now becomes, $|E_c^{out}| = \sum_{u \in c} \sum_{\substack{c' \in C \\ c' \neq c \\ v \in c'}} f(\alpha_{uc}, \alpha_{vc'}) A_{uv}$. $|E|$, which represents the total number of edges in the network is, $|E| = \frac{1}{2} \sum_{uv} A_{uv}$. The term $f(\alpha_{uc}, \alpha_{vc})$ is a function of the belonging coefficients $\alpha_{uc}$ and $\alpha_{vc}$. The basic difference between the commonly used extensions of modularity is observed in the way they define the belonging coefficients and the function $f(\alpha_{uc}, \alpha_{vc})$.

As mentioned earlier, the belonging coefficients are defined based on the ideas behind crisp and fuzzy assignment of nodes to communities. In a crisp assignment, a node belongs equally to all the communities it is assigned to. Using this idea, one formulation of the belonging coefficient $\alpha_{vc}$ that quantifies the association of node $v$ to a community $c$ is

$$a_{vc} = \frac{1}{O_v},$$

where $O_v$ is the total number of communities the node $v$ belongs to. Therefore, this assigns the same value to elements in the belonging coefficients vector $[\alpha_{vc_1}, \alpha_{vc_2}, \alpha_{vc_3}, ..., \alpha_{vc_k}]$ of a node $v$. In a fuzzy assignment, the nodes do not belong equally to assigned communities, and there are varying strengths associated with every assignment. Based on this notion, one way of defining the belonging coefficient $\alpha_{vc}$ is

$$\alpha_{vc} = \frac{\sum_{w \in c} A_{vw}}{\sum_{c' \in C_v} \sum_{w \in c'} A_{vw}}.$$

$C_v$ is the set of communities the node $v$ belongs to. Clearly, this formulation is the ratio of the number of internal edges a node has in the community to the total number of internal edges the node has across all the communities in the cover. These values could be different for different communties the node is assigned to, and hence the elements in the belonging coefficients vector $[\alpha_{vc_1}, \alpha_{vc_2}, \alpha_{vc_3}, ..., \alpha_{vc_k}]$ quantify how strongly a node connects to each community in the cover.

Given a definition for the belonging coefficients $\alpha_{uc}$ and $\alpha_{vc}$ of nodes $u$ and $v$ to a community $c$, the function $f(\alpha_{uc}, \alpha_{vc})$ is usually defined as either the product or average of the corresponding coefficients. That is, $f(\alpha_{uc}, \alpha_{vc})$ can be a product of the belonging coefficients

$$f(\alpha_{uc}, \alpha_{vc}) = \alpha_{uc}\alpha_{vc}.$$

Or it can be the average of the belonging coefficients involved

$$f(\alpha_{uc}, \alpha_{vc}) = \frac{\alpha_{uc} + \alpha_{vc}}{2}.$$

With all these and more definitions available for belonging coefficients and function $f$, many different versions of extended modularity are proposed and widely used. In the following section, I explain how an ideal extended modularity is selected in our case to compare the results of IDLE, GCE and SLPA on real networks.

### 5.2.2 Selecting an Ideal Extended Modularity

As discussed in Section 5.2.1, there are many definitions of extended modularity and it is important to select a formulation that best reflects the quality of detected overlapping communities in a network. Given the two definitions for belonging coefficients and two way to define the function $f$, there are four combinations that one can apply in $Q_{ov}$. To understand the behavior of each of these definitions of $Q_{ov}$, I apply them to networks where the ground-truth overlapping community structure is known. For this analysis, I randomly sample some LFR benchmark networks that are used in Section 5.1. I select six networks, each with different properties. All the networks contain $N = 5000$ nodes and contain large communities with sizes in the range $b = [20, 100]$. They differ in the number of overlapping nodes ($O_n$) and the number of memberships for each overlapping node ($O_m$). The properties of the six networks are as follows:

1. $G_{O_n=10\%,O_m=2}$: A network with fewer overlapping nodes ($O_n = 10\%$) and low overlapping memberships ($O_m = 2$)
2. $G_{O_n=10\%,O_m=5}$: A network with fewer overlapping nodes ($O_n = 10\%$) and moderate overlapping memberships ($O_m = 5$)
3. $G_{O_n=10\%,O_m=8}$: A network with fewer overlapping nodes ($O_n = 10\%$) and high overlapping memberships ($O_m = 8$)
4. $G_{O_n=50\%,O_m=2}$: A network with many overlapping nodes ($O_n = 50\%$) and lower overlapping memberships ($O_m = 2$)
5. $G_{O_n=50\%,O_m=5}$: A network with many overlapping nodes ($O_n = 50\%$) and moderate overlapping memberships ($O_m = 2$)
6. $G_{O_n=50\%,O_m=8}$: A network with many overlapping nodes ($O_n = 50\%$) and high overlapping memberships ($O_m = 8$)

The most suitable formulation of modularity should best capture the quality of overlapping communities present in these networks. To compare results, I also run the Louvain community detection [13] on these networks to get a disjoint community structure of them. I presume that the best extended modularity measure would acknowledge the ground-truth community structure by assigning higher modularity to it.

*Modularity density* $Q_{ds}$ is another measure of importance that was introduced by Chen et al. [19, 20] to address the tendency of Newman's modularity to sometimes prefer small communities over large ones and in some cases large communities over small ones.The latter tendency is referred to as the resolution limit problem in the literature [26]. Modularity density combines a *Split Penalty* and *Community Density* with modularity to construct a measure that is sensitive to community sizes in a network. Split penalty measures the fraction of edges that connect nodes in different communities and it is given for undirected networks as

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

$|E_{c,c'}|$ is the number of edges connecting nodes in community $c$ to nodes in community $c'$. In the case of weighted networks, this becomes the weighted sum of the corresponding edges. Subtracting the split penalty from modularity addresses the issue of favoring small communities over large ones. Community density takes into account both the number of nodes and edges in communities, and, to mitigate the resolution limit problem of modularity, Chen et al. [20] introduce this term into modularity density. Internal Community density of a given community ($d_c$) and pair-wise density between communities ($d_{c,c'}$) are given as

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

$$d_{c,c'} = \frac{|E_{c,c'}|}{|c||c'|}.$$

Both $|E_c^{in}|$ and $|E_{c,c'}|$ are defined to be unweighted even for weighted networks. Taking into account both the split penalty and the community densities, modularity density is defined as follows

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

Chen et al. [20] prove that modularity density solves the resolution limit problem. Though modularity density $Q_{ds}$ was originally proposed for disjoint community detection, they also propose an extended modularity density for overlapping community detection defined as

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

where,

$$d_c = \frac{2|E_c^{in}|}{\sum_{u,v \in c, u \neq v} f(\alpha_{uc}, \alpha_{vc})},$$

$$d_{c,c'} = \frac{|E_{c,c'}|}{\sum_{u \in c, v \in c'} f(\alpha_{uc}, \alpha_{vc'})},$$

$$|E_c^{in}| = \frac{1}{2} \sum_{u,v \in c} f(\alpha_{uc}, \alpha_{vc}) A_{uv},$$

$$|E_c^{out}| = \sum_{u \in c} \sum_{\substack{c' \in C \\ c' \neq c \\ v \in c'}} f(\alpha_{uc}, \alpha_{vc'}) A_{uv},$$

$$|E_{c,c'}| = \sum_{u \in c, v \in c'} f(\alpha_{uc}, \alpha_{vc'}) A_{uv},$$

$$|E| = \frac{1}{2} \sum_{uv} A_{uv}.$$

The values $\alpha_{uc}$, $\alpha_{vc}$ and $f(\alpha_{uc}, \alpha_{vc})$ are the same as the ones described for extended modularity. In the following analysis, I present the values for both extended modularity and extended modularity density obtained for the ground-truth cover and Louvain partition of the six networks using the four combinations of belonging coefficients and function $f$ described earlier.

**Case 1**

In Case 1, the extended modularity $Q_{ov}$ and extended modularity density $Q_{ds}^{ov}$ use

$$\alpha_{vc} = \frac{1}{O_v},$$

$$f(\alpha_{uc}, \alpha_{vc}) = \alpha_{uc}\alpha_{vc}.$$

I first present the values of extended modularity and extended modularity density for the ground-truth cover and Louvain partition. Further, I also show these values for the covers produced by IDLE, GCE and SLPA on these networks. The parameters for these algorithms remain the same as the default values used for our experiments on synthetic networks. For SLPA, the value reported is the maximum extended modularity and the corresponding extended modularity density obtained from among the runs with different $r$ values.

Table 5.1: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 1, for ground-truth cover and Louvain partition

| | Ground-truth Cover | | Louvain Partition | |
|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | 0.6193 | 0.0873 | **0.6363** | 0.0407 |
| $G_{O_n=10\%,O_m=5}$ | 0.5859 | 0.1008 | **0.6044** | 0.0448 |
| $G_{O_n=10\%,O_m=8}$ | 0.5680 | 0.1175 | **0.5886** | 0.0494 |
| $G_{O_n=50\%,O_m=2}$ | 0.3835 | 0.0481 | **0.4567** | 0.0157 |
| $G_{O_n=50\%,O_m=5}$ | 0.2344 | 0.0343 | **0.3488** | 0.0083 |
| $G_{O_n=50\%,O_m=8}$ | 0.1808 | 0.0319 | **0.3189** | 0.0063 |

Table 5.2: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 1, for covers from IDLE, GCE and SLPA

| | IDLE | | GCE | | SLPA | |
|---|---|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | 0.6217 | 0.0877 | 0.5968 | 0.0902 | 0.6164 | 0.0857 |
| $G_{O_n=10\%,O_m=5}$ | 0.5902 | 0.1019 | 0.5480 | 0.1147 | 0.5875 | 0.1010 |
| $G_{O_n=10\%,O_m=8}$ | 0.5753 | 0.1192 | 0.5425 | 0.1356 | 0.5717 | 0.1125 |
| $G_{O_n=50\%,O_m=2}$ | 0.3566 | 0.0428 | 0.3256 | 0.0430 | 0.4132 | 0.0480 |
| $G_{O_n=50\%,O_m=5}$ | 0.2390 | 0.0259 | 0.1646 | 0.0263 | 0.2959 | 0.0380 |
| $G_{O_n=50\%,O_m=8}$ | 0.2017 | 0.0219 | 0.1235 | 0.0230 | 0.0378 | 0.0085 |

One can observe that the extended modularity is highest for disjoint partition obtained using Louvain community detection algorithm, and the corresponding value for extended modularity density is comparatively lower than what is obtained for ground-truth covers. Similarly, the highest values among the covers from other algorithms is obtained for SLPA and these values are higher than what is obtained for ground-truth covers. The SLPA threshold $r$ corresponding to these values of $Q_{ov}$ and $Q_{ds}^{ov}$ is either 0.45 or 0.5 From [74], SLPA gives disjoint communities when $r \geq 0.5$. This behavior is observed because the total number of communities a node belongs to, $O_i$, is inversely proportional to this formulation of extended modularity. Hence the fewer communities a node belongs to, the better is its contribution to $Q_{ov}$. This results in higher values for disjoint community structure of networks.

### Case 2

In Case 2, the extended modularity $Q_{ov}$ and extended modularity density $Q_{ds}^{ov}$ take

$$\alpha_{vc} = \frac{1}{O_v},$$

$$f(\alpha_{uc}, \alpha_{vc}) = \frac{\alpha_{uc} + \alpha_{vc}}{2}.$$

Table 5.3: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 2, for ground-truth cover and Louvain partition

| | Ground-truth Cover | | Louvain Partition | |
|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | **0.6504** | 0.0878 | 0.6363 | 0.0407 |
| $G_{O_n=10\%,O_m=5}$ | **0.6296** | 0.0842 | 0.6044 | 0.0448 |
| $G_{O_n=10\%,O_m=8}$ | **0.6158** | 0.0830 | 0.5886 | 0.0494 |
| $G_{O_n=50\%,O_m=2}$ | **0.5073** | 0.0564 | 0.4567 | 0.0157 |
| $G_{O_n=50\%,O_m=5}$ | **0.3886** | 0.0308 | 0.3488 | 0.0083 |
| $G_{O_n=50\%,O_m=8}$ | **0.3306** | 0.0208 | 0.3189 | 0.0063 |

Table 5.4: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 2, for covers from IDLE, GCE and SLPA

| | IDLE | | GCE | | SLPA | |
|---|---|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | 0.6440 | 0.0851 | 0.6092 | 0.0905 | 0.6290 | 0.0809 |
| $G_{O_n=10\%,O_m=5}$ | 0.6369 | 0.0884 | 0.5500 | 0.1145 | 0.5970 | 0.0962 |
| $G_{O_n=10\%,O_m=8}$ | 0.6288 | 0.0958 | 0.5438 | 0.1352 | 0.5785 | 0.0974 |
| $G_{O_n=50\%,O_m=2}$ | 0.4913 | 0.0383 | 0.4232 | 0.0402 | 0.4440 | 0.0417 |
| $G_{O_n=50\%,O_m=5}$ | 0.3653 | 0.0217 | 0.2303 | 0.0221 | 0.3217 | 0.0309 |
| $G_{O_n=50\%,O_m=8}$ | 0.3069 | 0.0160 | 0.1583 | 0.0204 | 0.0367 | 0.0086 |

As we can see from Table 5.3, the highest values for $Q_{ov}$ and $Q_{ds}^{ov}$ are obtained for ground-truth covers of the networks. One can see that the values for Louvain partition are the same as those observed in Case 1. This is because, for any formulation of belonging coefficient and function $f$, those values evaluate to 1 since a node only belongs to a single community. This reduces extended modularity to Newman's modularity, and extended modularity density to the one for disjoint communities. This can be observed even in Case 3 and Case 4 discussed below. The values obtained for Louvain partition in this case are lower than the corresponding values obtained for ground-truth cover. From Table 5.4, the extended modularity and extended modularity density of covers produced by IDLE, GCE and SLPA can be studied.

### Case 3

In Case 3, the extended modularity $Q_{ov}$ and extended modularity density $Q_{ds}^{ov}$ take

$$\alpha_{vc} = \frac{\sum_{w \in c} A_{vw}}{\sum_{c' \in C_v} \sum_{w \in c'} A_{vw}},$$

$$f(\alpha_{uc}, \alpha_{vc}) = \alpha_{uc}\alpha_{vc}.$$

Table 5.5: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 3, for ground-truth cover and Louvain partition

| | Ground-truth Cover | | Louvain Partition | |
|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | 0.6205 | 0.0876 | **0.6363** | 0.0407 |
| $G_{O_n=10\%,O_m=5}$ | 0.5899 | 0.1022 | **0.6044** | 0.0448 |
| $G_{O_n=10\%,O_m=8}$ | 0.5742 | 0.1203 | **0.5886** | 0.0494 |
| $G_{O_n=50\%,O_m=2}$ | 0.3886 | 0.0493 | **0.4567** | 0.0157 |
| $G_{O_n=50\%,O_m=5}$ | 0.2485 | 0.0401 | **0.3488** | 0.0083 |
| $G_{O_n=50\%,O_m=8}$ | 0.2006 | 0.0423 | **0.3189** | 0.0063 |

Table 5.6: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 3, for covers from IDLE, GCE and SLPA

| | IDLE | | GCE | | SLPA | |
|---|---|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | 0.6226 | 0.0881 | 0.5979 | 0.0903 | 0.6164 | 0.0857 |
| $G_{O_n=10\%,O_m=5}$ | 0.5909 | 0.1022 | 0.5487 | 0.1149 | 0.5875 | 0.1010 |
| $G_{O_n=10\%,O_m=8}$ | 0.5756 | 0.1194 | 0.5429 | 0.1356 | 0.5717 | 0.1125 |
| $G_{O_n=50\%,O_m=2}$ | 0.3718 | 0.0460 | 0.3372 | 0.0455 | 0.4132 | 0.0480 |
| $G_{O_n=50\%,O_m=5}$ | 0.2481 | 0.0277 | 0.1729 | 0.0284 | 0.2959 | 0.0380 |
| $G_{O_n=50\%,O_m=8}$ | 0.2082 | 0.0232 | 0.1282 | 0.0247 | 0.0367 | 0.0086 |

The results obtained for Case 3 are very similar to the $Q_{ov}$ and $Q_{ds}^{ov}$ values obtained in Case 1. Similar to Case 1, the modularity obtained for Louvain Partitions form the upper bound for extended modularity values obtained for all the other covers of the same network.

### Case 4

In Case 4, the extended modularity $Q_{ov}$ and extended modularity density $Q_{ds}^{ov}$ use

$$\alpha_{vc} = \frac{\sum_{w \in c} A_{vw}}{\sum_{c' \in C_v} \sum_{w \in c'} A_{vw}},$$

$$f(\alpha_{uc}, \alpha_{vc}) = \frac{\alpha_{uc} + \alpha_{vc}}{2}.$$

Table 5.7: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 4, for ground-truth cover and Louvain partition

| | Ground-truth Cover | | Louvain Partition | |
|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | **0.6511** | 0.0880 | 0.6363 | 0.0407 |
| $G_{O_n=10\%,O_m=5}$ | **0.6318** | 0.0846 | 0.6044 | 0.0448 |
| $G_{O_n=10\%,O_m=8}$ | **0.6191** | 0.0837 | 0.5886 | 0.0494 |
| $G_{O_n=50\%,O_m=2}$ | **0.5106** | 0.0571 | 0.4567 | 0.0157 |
| $G_{O_n=50\%,O_m=5}$ | **0.3992** | 0.0324 | 0.3488 | 0.0083 |
| $G_{O_n=50\%,O_m=8}$ | **0.3476** | 0.0228 | 0.3189 | 0.0063 |

Table 5.8: Table showing $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 4, for covers from IDLE, GCE and SLPA

| | IDLE | | GCE | | SLPA | |
|---|---|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| $G_{O_n=10\%,O_m=2}$ | 0.6446 | 0.0853 | 0.6097 | 0.0906 | 0.6358 | 0.0778 |
| $G_{O_n=10\%,O_m=5}$ | 0.6374 | 0.0885 | 0.5504 | 0.1145 | 0.6019 | 0.0919 |
| $G_{O_n=10\%,O_m=8}$ | 0.6292 | 0.0959 | 0.5440 | 0.1352 | 0.5833 | 0.0984 |
| $G_{O_n=50\%,O_m=2}$ | 0.5010 | 0.0396 | 0.4299 | 0.0412 | 0.4546 | 0.0385 |
| $G_{O_n=50\%,O_m=5}$ | 0.3720 | 0.0225 | 0.2347 | 0.0228 | 0.3313 | 0.0321 |
| $G_{O_n=50\%,O_m=8}$ | 0.3114 | 0.0166 | 0.1609 | 0.0211 | 0.0367 | 0.0086 |

The values for $Q_{ov}$ and $Q_{ds}^{ov}$ in Case 4 shown in Table 5.8 is very close to the values obtained in Case 2. Also, it is evident that extended modularity is the best for ground-truth covers in this case. It is also intuitive to calculate the strength of association of nodes to communities based on how tightly they are connected to those communities. Due to these reasons, I have adopted $Q_{ov}$ and $Q_{ds}^{ov}$ with belonging coefficient $\alpha_{vc} = \frac{\sum_{w \in c} A_{vw}}{\sum_{c' \in C_v} \sum_{w \in c'} A_{vw}}$ and function $f(\alpha_{uc}, \alpha_{vc}) = \frac{\alpha_{uc}+\alpha_{vc}}{2}$ to measure the quality of overlapping communities detected by IDLE, GCE and SLPA on real networks in Section 5.2.3.

### 5.2.3 Results on Real Networks

In this section, I measure the quality of covers generated by IDLE, GCE and SLPA for real networks that are studied in [73] along with a few others. I believe this helps the reader understand the strengths of IDLE, since the networks are chosen by other parties. The properties of these networks along with their sources are presented in Table 5.9. The data files for all the networks are obtained from [47] and [52].

IDLE, GCE and SLPA are run on each of these networks, and they generate covers for them. The extended modularity $Q_{ov}$ and extended modularity density $Q_{ds}^{ov}$ are used to report the quality of these covers. The belonging coefficients and function $f$ used in $Q_{ov}$ and $Q_{ds}^{ov}$ take the form

Table 5.9: Table showing the properties of real networks used in experiments

| Network | #Nodes, $\lvert V\rvert$ | #Edges, $\lvert E\rvert$ | Source |
|---|---|---|---|
| Karate | 34 | 78 | Zachary [76] |
| Football | 115 | 613 | Girvan and Newman [27] |
| Les Miserables | 77 | 384 | Knuth [38] |
| Dolphins | 62 | 159 | Lusseau et al. [49] |
| CA-GrQc | 4158 | 13428 | Leskovec et al. [46] |
| PGP | 10680 | 24316 | Boguna et al. [14] |
| CA-CondMat | 21363 | 91342 | Leskovec et al. [46] |
| Email | 33696 | 180811 | Klimt and Yang [37] |
| P2P | 62561 | 147878 | Ripeanu et al. [60] |
| Epinions | 75877 | 405739 | Richardson et al. [59] |
| Amazon | 262111 | 1234877 | Leskovec et al. [45] |

discussed in Case 4:

$$\alpha_{vc} = \frac{\sum_{w \in c} A_{vw}}{\sum_{c' \in C_v} \sum_{w \in c'} A_{vw}}$$

$$f(\alpha_{uc}, \alpha_{vc}) = \frac{\alpha_{uc} + \alpha_{vc}}{2}.$$

In addition to the covers obtained, I generate the Louvain partition for each of these networks. The $Q_{ov}$ and $Q_{ds}^{ov}$ values are obtained for the partitions as well and presented along with the other results. This would give the reader an overall picture of the performance of different techniques in detecting community structures in networks. Table 5.10 presents the results measured as extended modularity and extended modularity density of the covers generated for the real networks and Figure 5.15 presents the runtime measured for the algorithms on the same networks in the same order as shown in Table 5.10. As already noted, the runtime values cannot be directly compared since their implementations are provided in different languages. However, we see that the runtime values for a given network show very small differences for the three algorithms. For SLPA, though the algorithm should be run for multiple $r$ values before we can decide the best cover, the runtime shown corresponds to the time taken for a single $r$ value.

Table 5.10: Table showing the values of $Q_{ov}$ and $Q_{ds}^{ov}$ for real networks

| | IDLE | | GCE | | SLPA | | Louvain Partition | |
|---|---|---|---|---|---|---|---|---|
| Network | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ | $Q_{ov}$ | $Q_{ds}^{ov}$ |
| Karate | **0.3791** | 0.2188 | **0.3299** | 0.1623 | **0.1328** | 0.1643 | **0.4188** | 0.2281 |
| Football | **0.6005** | 0.4909 | **0.5945** | 0.4929 | **0.5770** | 0.3331 | **0.6041** | 0.4173 |
| Les Miserables | **0.4904** | 0.2788 | **0.5200** | 0.2745 | **0.5307** | 0.2850 | **0.5435** | 0.3252 |
| Dolphins | **0.4999** | 0.1847 | **0.4823** | 0.1760 | **0.3787** | 0.1362 | **0.5233** | 0.1994 |
| CA-GrQc | **0.7921** | 0.3015 | **0.7117** | 0.3132 | **0.8037** | 0.2539 | **0.8457** | 0.1413 |
| PGP | **0.8188** | 0.1524 | **0.5998** | 0.1182 | **0.8158** | 0.1380 | **0.8829** | 0.0309 |
| CA-CondMat | **0.6812** | 0.1441 | **0.6322** | 0.1609 | **0.6591** | 0.1489 | **0.7230** | 0.0129 |
| Email | **0.3062** | 0.0315 | **0.5161** | 0.0284 | **0.3207** | 0.0387 | **0.6063** | 0.0141 |
| P2P | **0.5508** | 0.0622 | **0.0023** | 0.0002 | **0.4542** | 0.0420 | **0.4985** | 0.0009 |
| Epinions | **0.2079** | 0.0105 | **0.0651** | 0.0032 | **0.0647** | 0.0096 | **0.4411** | 0.0054 |
| Amazon | **0.7518** | 0.2554 | **0.6267** | 0.2782 | **0.7638** | 0.1906 | **0.9014** | 0.0044 |

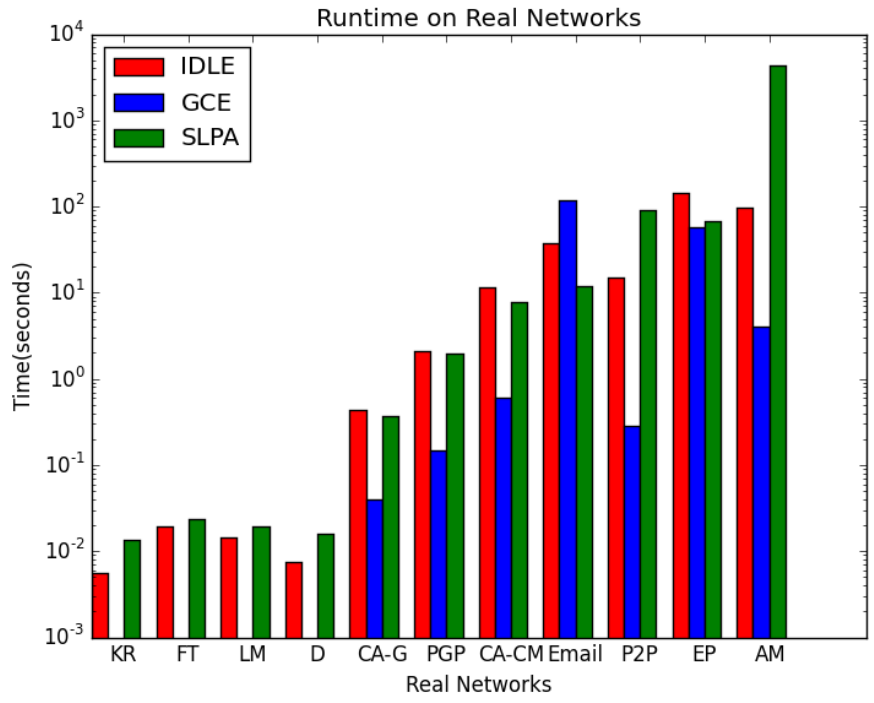


Figure 5.15: Time taken (in seconds) by IDLE, GCE and SLPA to produce covers on the real networks used in experiments.

From the results shown in Table 5.10, one can see that there are some networks where all the algorithms produce good modularity covers and some where all three do not perform well. In some networks (especially the small ones like Karate, Les Miserables and Dolphins), the disjoint Louvain partition produces the best extended modularity combined with the best extended modularity density. In such cases, one understands that the network structure clearly favors a partition over a cover. One can also observe that Louvain partition attains the highest modularity even for large networks. But the corresponding modularity density is very small compared to the ones obtained for the covers. This allows us to infer that a cover is better in such cases. However, in the case of large networks, we can see that IDLE consistently produces covers with high extended modularity combined with a high extended modularity density. From the experiments in Section 5.1, we see that IDLE is also capable of detecting overlapping nodes and their memberships much better than both GCE and SLPA can. All these point to the fact that IDLE is one of the few overlapping community detection algorithms that can be considered state-of-the-art.

# Chapter 6

# Conclusions and Future Work

In this thesis, I motive the problem of community detection by pointing out the insights that can be achieved through community structure analysis. Overlapping community detection adds another dimension of information to this analysis, and hence is considered an important tool in the study of complex networks. Though there are quite a number of overlapping community detection algorithms already proposed for complex networks, the results obtained using them show room for significant improvement of detection quality. To achieve this, I propose a local expansion and optimization based overlapping community detection algorithm called IDLE. In this algorithm, I adopt an interesting idea from influential spreaders research to efficiently identify seed nodes. I show how conductance is related to the fitness function that is commonly used for greedy expansion and local community detection and adopt it for IDLE. One important feature of this algorithm that contributes to improved results is its explicit effort to prune out weakly connected nodes from communities and also detect overlapping nodes and their memberships. I compare the accuracy of IDLE with two algorithms, GCE and SLPA, that are considered state-of-the-art in overlapping community detection. Through extensive experiments conducted using synthetic and real networks, I demonstrate that IDLE indeed results in an improved detection of overlapping communities in complex networks.

In this work, I also present a detailed study of the different modularity measures that are currently used to quantify the performance of algorithms in the absence of the ground-truth overlapping community structure. I show why certain formulations of this modularity are preferred over others and combine those with another measure called the modularity density, to present an accurate picture of the performance of different algorithms in overlapping community detection.

As one can see from the results presented, the performance of all community detection algorithms, including IDLE, drop drastically when the networks have high overlapping density and high overlapping diversity. IDLE could be improved further to detect better overlapping communities in such networks.

In all experiments conducted using IDLE, the phase 2 detection of overlapping nodes is disabled due to its dependence on the $\gamma$ value required from the user. The algorithm could be improved so it learns this value from the network structure itself, and thereby achieve an enhanced detection of overlapping nodes.

The current implementation of IDLE uses basic graph operations provided as part of the networkx library and hence is implemented in Python. Its run-time, therefore, cannot be directly compared to other algorithms like GCE that are implemented in compiled languages such as C or C++. It would be useful to implement IDLE in a language like C or C++ to achieve a natural speedup of the implementation.

# Bibliography

[1] Lanet-vi: Large networks visualization tool. `http://lanet-vi.soic.indiana.edu/lanetvi.php`, 2016 (Accessed: January 3, 2016).

[2] Yong-Yeol Ahn, James P. Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307):761–764, 2010.

[3] William Aiello, Fan Chung, and Linyuan Lu. A random graph model for massive graphs. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 171–180, 2000.

[4] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.

[5] Barabási Albert-László. *Linked: The New Science of Networks*. Perseus, 2002.

[6] J. Ignacio Alvarez-Hamelin, Luca Dall'Asta, Alain Barrat, and Alessandro Vespignani. Large scale networks fingerprinting and visualization using the $k$-core decomposition. In *Advances in Neural Information Processing Systems*, pages 41–50, 2005.

[7] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, 56(2):5(1–37), 2009.

[8] Brian Ball, Brian Karrer, and M.E.J. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3):036103(1–13), 2011.

[9] Albert-László Barabási, Réka Albert, and Hawoong Jeong. Scale-free characteristics of random networks: The topology of the World-Wide Web. *Physica A: Statistical Mechanics and Its Applications*, 281(1):69–77, 2000.

[10] Albert-László Barabási, Hawoong Jeong, Zoltan Néda, Erzsebet Ravasz, Andras Schubert, and Tamas Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and Its Applications*, 311(3):590–614, 2002.

[11] Vladimir Batagelj and Matjaz Zaversnik. An O($m$) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*, 2003.

[12] Jeffrey Baumes, Mark K. Goldberg, Mukkai S. Krishnamoorthy, Malik Magdon-Ismail, and Nathan Preston. Finding communities by clustering a graph into overlapping subgraphs. *Proceedings of the IADIS International Conference on Applied Computing*, 5:97–104, 2005.

[13] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008(1–12), 2008.

[14] Marián Boguñá, Romualdo Pastor-Satorras, Albert Díaz-Guilera, and Alex Arenas. Models of social networks based on social distance attachment. *Physical Review E*, 70(5):056122(1–8), 2004.

[15] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the web. *Computer Networks*, 33(1):309–320, 2000.

[16] Coen Bron and Joep Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[17] Duanbing Chen, Mingsheng Shang, Zehua Lv, and Yan Fu. Detecting overlapping communities of weighted networks via a local algorithm. *Physica A: Statistical Mechanics and Its Applications*, 389(19):4177–4187, 2010.

[18] Mingming Chen, Konstantin Kuzmin, and Boleslaw K. Szymanski. Extension of modularity density for overlapping community structure. In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 856–863. IEEE, 2014.

[19] Mingming Chen, Thin Nguyen, and Boleslaw K. Szymanski. On measuring the quality of a network community structure. In *2013 International Conference on Social Computing (SocialCom)*, pages 122–127. IEEE, 2013.

[20] Mingming Chen, Tommy Nguyen, and Boleslaw K. Szymanski. A new metric for quality of network community structure. *ASE Human Journal*, 2(4):226–240, 2013.

[21] Linda M. Collins and Clyde W. Dent. Omega: A general formulation of the Rand index of cluster recovery suitable for non-disjoint solutions. *Multivariate Behavioral Research*, 23(2):231–242, 1988.

[22] Leon Danon, Albert Diaz-Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008(1–10), 2005.

[23] Gerald F. Davis and Henrich R. Greve. Corporate elite networks and governance changes in the 1980s. *American Journal of Sociology*, 103(1):1–37, 1997.

[24] Hristo N. Djidjev. A scalable multilevel algorithm for graph clustering and community structure detection. In *Algorithms and models for the web-graph*, pages 117–128. Springer, 2006.

[25] Sergey N. Dorogovtsev and Jose F.F. Mendes. Evolution of networks. *Advances in Physics*, 51(4):1079–1187, 2002.

[26] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[27] Michelle Girvan and Mark E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[28] Steve Gregory. Fuzzy overlapping communities in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(02):P02017, 2011.

[29] Frank Havemann, Michael Heinz, Alexander Struck, and Jochen Gläser. Identification of overlapping communities and their hierarchy by locally calculating community-changing resolution levels. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(01):P01023(1–23), 2011.

[30] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[31] Takashi Ito, Tomoko Chiba, Ritsuko Ozawa, Mikio Yoshida, Masahira Hattori, and Yoshiyuki Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proceedings of the National Academy of Sciences*, 98(8):4569–4574, 2001.

[32] Hawoong Jeong, Bálint Tombor, Réka Albert, Zoltan N. Oltvai, and A-L. Barabási. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[33] Lucas G.S. Jeub, Prakash Balachandran, Mason A. Porter, Peter J. Mucha, and Michael W. Mahoney. Think locally, act locally: Detection of small, medium-sized, and large communities in large networks. *Physical Review E*, 91(1):012821(1–29), 2015.

[34] Stephen Kelley. *The existence and discovery of overlapping communities in large-scale networks*. Ph.D. Thesis, Rensselaer Polytechnic Institute, Troy, NY, 2009.

[35] Maksim Kitsak, Lazaros K. Gallos, Shlomo Havlin, Fredrik Liljeros, Lev Muchnik, H. Eugene Stanley, and Hernán A. Makse. Identification of influential spreaders in complex networks. *Nature Physics*, 6(11):888–893, 2010.

[36] Jon M. Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew S. Tomkins. The Web as a graph: Measurements, models, and methods. In *Computing and Combinatorics*, pages 1–17. Springer, 1999.

[37] Bryan Klimt and Yiming Yang. Introducing the Enron corpus. In *First Conference on Email and Anti-Spam (CEAS)*, 2004.

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

[39] Renaud Lambiotte, Vincent D. Blondel, Cristobald De Kerchove, Etienne Huens, Christophe Prieur, Zbigniew Smoreda, and Paul Van Dooren. Geographical dispersal of mobile communication networks. *Physica A: Statistical Mechanics and Its Applications*, 387(21):5317–5325, 2008.

[40] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1):016118(1–8), 2009.

[41] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, 2009.

[42] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110(1–5), 2008.

[43] Conrad Lee, Aaron McDaid, Fergal Reid, and Neil J. Hurley. Detecting highly overlapping community structure by greedy clique expansion. In *Paper presented at the 4th SNA-KDD Workshop10 (SNA-KDD10), held in conjunction with The 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2010), July 25, 2010, Washington, DC USA*, 2010.

[44] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, 46(6):787–832, 1999.

[45] Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5(1–39), 2007.

[46] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2(1–41), 2007.

[47] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[48] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

[49] David Lusseau, Karsten Schneider, Oliver J. Boisseau, Patti Haase, Elisabeth Slooten, and Steve M. Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.

[50] Stanley Milgram. The small world problem. *Psychology Today*, 2(1):60–67, 1967.

[51] Tamás Nepusz, Andrea Petróczi, László Négyessy, and Fülöp Bazsó. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 77(1):016107(1–12), 2008.

[52] Mark E.J. Newman. `http://www-personal.umich.edu/~mejn/netdata`.

[53] Mark E.J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

[54] Mark E.J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

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

[56] Vincenzo Nicosia, Giuseppe Mangioni, Vincenza Carchiolo, and Michele Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03024(1–23), 2009.

[57] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[58] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005*, pages 284–293. Springer, 2005.

[59] Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. Trust management for the semantic web. In *The Semantic Web-ISWC 2003*, pages 351–368. Springer, 2003.

[60] Matei Ripeanu, Adriana Iamnitchi, and Ian Foster. Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing*, 6(1):50–57, 2002.

[61] Erin N. Sawardecker, Marta Sales-Pardo, and Luıs A. Nunes Amaral. Detection of node group membership in networks with group overlap. *The European Physical Journal B*, 67(3):277–284, 2009.

[62] Stephen B. Seidman. Network structure and minimum degree. *Social Networks*, 5(3):269–287, 1983.

[63] Hua-Wei Shen, Xue-Qi Cheng, and Jia-Feng Guo. Quantifying and identifying the overlapping community structure in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(07):P07042(1–16), 2009.

[64] Huawei Shen, Xueqi Cheng, Kai Cai, and Mao-Bin Hu. Detect overlapping and hierarchical community structure in networks. *Physica A: Statistical Mechanics and Its Applications*, 388(8):1706–1712, 2009.

[65] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[66] Alistair Sinclair and Mark Jerrum. Approximate counting, uniform generation and rapidly mixing markov chains. *Information and Computation*, 82(1):93–133, 1989.

[67] Steven H Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.

[68] Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, 363(1):28–42, 2006.

[69] Peter Uetz, Loic Giot, Gerard Cagney, Traci A. Mansfield, Richard S. Judson, James R. Knight, Daniel Lockshon, Vaibhav Narayan, Maithreyan Srinivasan, Pascale Pochart, et al. A comprehensive analysis of protein–protein interactions in *Saccharomyces cerevisiae*. *Nature*, 403(6770):623–627, 2000.

[70] Stijn Marinus Van Dongen. Graph clustering by flow simulation. *Ph.D. Thesis, University of Utrecht, The Netherlands.*, 2000.

[71] Andreas Wagner and David A. Fell. The small world inside large metabolic networks. *Proceedings of the Royal Society of London B: Biological Sciences*, 268(1478):1803–1810, 2001.

[72] Duncan J. Watts. Networks, dynamics, and the small-world phenomenon 1. *American Journal of Sociology*, 105(2):493–527, 1999.

[73] Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Computing Surveys*, 45(4):43, 2013.

[74] Jierui Xie and Boleslaw K Szymanski. Towards linear time overlapping community detection in social networks. In *Advances in Knowledge Discovery and Data Mining*, pages 25–36. Springer, 2012.

[75] Jierui Xie, Boleslaw K. Szymanski, and Xiaoming Liu. SLPA: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In *2011 IEEE 11th International Conference on Data Mining Workshops (ICDMW)*, pages 344–349. IEEE, 2011.

[76] Wayne W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977.

[77] Shihua Zhang, Rui-Sheng Wang, and Xiang-Sun Zhang. Identification of overlapping community structure in complex networks using fuzzy $c$-means clustering. *Physica A: Statistical Mechanics and Its Applications*, 374(1):483–490, 2007.