2019

COMMUNITY DETECTION IN COMPLEX NETWORKS AND APPLICATION TO DENSE WIRELESS SENSOR NETWORKS LOCALIZATION

Sakineh Yazdanparast
Michigan Technological University, syazdanp@mtu.edu

COMMUNITY DETECTION IN COMPLEX NETWORKS AND APPLICATION TO DENSE WIRELESS SENSOR NETWORKS LOCALIZATION

By

Sakineh Yazdanparast

A DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

In Electrical Engineering

MICHIGAN TECHNOLOGICAL UNIVERSITY

2019

This dissertation has been approved in partial fulfillment of the requirements for the Degree of DOCTOR OF PHILOSOPHY in Electrical Engineering.

Department of Electrical and Computer Engineering

Dissertation Advisor:	*Dr. Timothy C. Havens*
Committee Member:	*Dr. Michael C. Roggemann*
Committee Member:	*Dr. Jeremy P. Bos*
Committee Member:	*Dr. Alexander E. Labovsky*
Department Chair:	*Dr. Daniel R. Fuhrmann*

Dedication

To Dr. Timothy C. Havens, Mohsen and Leah

who collectively are my greatest inspiration in my life.

Contents

List of Figures

List of Tables

Preface

Some chapters of this dissertation contain published material and unpublished research. The following list indicates which publications and unpublished work, including manuscripts that have been submitted but not yet accepted, were used along with notes on author contributions.

Chapter 2

S. Yazdanparast, T.C. Havens, "Modularity maximization using completely positive programming," Physica A: Statistical Mechanics and its Applications, Volume 471, 2017, Pages 20-32, ISSN 0378-4371, https://doi.org/10.1016/j.physa.2016.11.108 (See [4]).

S. Yazdanparast is the leading researcher in this work and is the corresponding author. The research was performed under the guidance of T.C. Havens.

Chapter 3

S.Yazdanparast, T.C. Havens,"Super Fast Community Detection via Hybrid Label Propagation", *In review, PHYSICAL REVIEW E covering statistical, nonlinear, biological, and soft matter physics on September 17 2018.*

S. Yazdanparast is the leading researcher in this work and is the corresponding author. The research was performed under the guidance of T.C. Havens.

Chapter 4

S. Yazdanparast, M. Jamalabdollahi, and T.C. Havens."Linear Time Community Detection by a Novel Modularity Gain Acceleration in Label Propagation," *In review, IEEE Transactions on Big Data, submitted on March 12 2019.*

The ideas presented in this paper are the result of discussions between all listed authors. S. Yazdanparast is the corresponding author for this paper and generated the experimental results. T.C. Havens and M. Jamalabdollahi contributed the theoretical background.

Chapter 5

S. Yazdanparast, T.C. Havens, and M. Jamalabdollahi (2019), "Overlapping Community Detection in Large-Scale Complex Networks via Fast Fuzzy Modularity Maximization," *In preparation, IEEE Transactions on Fuzzy Systems.*

S. Yazdanparast is the leading researcher in this work and is the corresponding author. The research was performed under the guidance of T.C. Havens, and M. Jamalabdollahi assisted with the code written for the experiments.

Chapter 6

S. Yazdanparast, T.C. Havens, and M. Jamalabdollahi (2019), "Range Free Anchor Selection in Wireless Sensor Networks via Community Detection," *In preparation, IEEE Transaction on Computational Social Sysems.*

S. Yazdanparast is the leading researcher in this work and is the corresponding author. The research was performed under the guidance of T.C. Havens. Some ideas in this work originated in conversations with M. Jamalabdollahi.

Acknowledgments

I would like to express my gratitude toward my Master and PhD advisor, Dr. Timothy C. Havens for his support and encouragement in my work. I could not have imagined having a better mentor for my Ph.D study. I never could have finished this dissertation without the guidance of him.

I wish to express my great appreciation to the Dr. Michael C. Roggemann , Dr. Jeremy P. Bos and Dr. Alexander E. Labovsky for being my thesis committee, and for their great advice on my research and evaluating my thesis.

I wish to extend special appreciation to the ECE department faculty, staff and friends for their support especially department chair, Dr. Daniel R. Fuhrmann.

A Heartfelt thanks to all supportive wonderful MTU members and friends. MTU for me is so much more than a university; it is a community that I truly feel a part of, a family and a real home, for international students.

I feel so proud to be an MTU graduate student. At the end, I would like to express my indebtedness to my family, my parent who have given me unlimited love and care during the completion of the thesis.

Abstract

Network analysis is applied in numerous researches. Features and characteristics of complex networks provide information associated with a network feature called community structure. Naturally, nodes with similar attributes will be more likely to form a community. Community detection is described as the process by which complex network data are analyzed to uncover organizational properties, and structure; and ultimately to enable extraction of useful information. Analysis of *Wireless Sensor Networks* (WSN) is considered as one of the most important categories of network analysis due to their enormous and emerging applications. Most WSN applications are location-aware, which entails precise localization of the deployed sensor nodes. However, localization of sensor nodes in very dense network is a challenging task. Among various challenges associated with localization of dense WSNs, anchor node selection is shown as a prominent open problem. Optimum anchor selection impacts overall sensor node localization in terms of accuracy and consumed energy. In this thesis, various approaches are developed to address both overlapping and non-overlapping community detection. The proposed approaches target small-size to very large-size networks in near linear time, which is important for very large, densely-connected networks. Performance of the proposed techniques are evaluated over real-world data sets with up to 10^6 nodes and syntactic networks via Newman's Modularity and *Normalized Mutual Information* (NMI). Moreover, the proposed community detection

approaches are extended to develop a novel criterion for range-free anchor selection in WSNs. Our approach uses novel objective functions based on nodes' community memberships to reveal a set of anchors among all available permutations of anchors-selection sets. The performance—the mean and variance of the localization error—of the proposed approach is evaluated for a variety of node deployment scenarios and compared with random anchor selection and the full-ranging approach. In order to study the effectiveness of our algorithm, the performance is evaluated over several simulations that randomly generate network configurations. By incorporating our proposed criteria, the accuracy of the position estimate is improved significantly relative to random anchor selection localization methods. Simulation results show that the proposed technique significantly improves both the accuracy and the precision of the location estimation.

Chapter 1

Introduction

Network analysis is applied over a variety of research topics such as social networks, transportation, anthropology, biology, economics, sociology, and bibliometrics studies [5, 6, 7]. Random deployment of multiple sensors over a wireless multi-hop link in a given area forms a network which is referred to as a *Wireless Sensor Network* (WSN). WSNs are considered as a prominent and well-studied category of networks.

WSNs are used for location-aware monitoring or detection of events or measurements, and for reporting of parameters or information. For a brief review of WSN applications one can refer to the following topics: environmental monitoring [8], search and rescue [9], health [10, 11] and target monitoring and tracking [12], road traffic monitoring [13], underground monitoring [14], disaster relief [15], structure health monitoring [16], etc.

1.1 Community Detection

Features and characteristics in complex networks provide information associated with a network feature called the community structure. A community in a network is often defined as a group of nodes—i.e., users—that have more concentrated connections or data in common among the group's members than with the rest of the network. Naturally, nodes with similar attributes will be more likely to form a community. Community detection is one of the most important problems in network analysis.

The process in which complex network data are analyzed in order to uncover organizational principles, properties, and structure of complex networks, and ultimately to enable extraction of useful information from them is called community detection. Community detection has attracted much attention in the past two decades. Community detection approaches can be divided into three main categories [17]: traditional methods [18, 19], divisive algorithms [20, 21, 22], and modularity-based methods [1, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33]. Traditional methods, such as graph partitioning and clustering, divide or merge clusters based on the similarity between nodes, whereas divisive algorithms are based on removing edges that connect nodes with lower similarity. Modularity based methods work by optimizing an objective function such as *modularity* introduced by Newman and Girvan [34]. Most of the recent works on community detection are based on maximization of the modularity objective function proposed by Newman [23, 24, 25, 26, 31]. Other works have proposed other objective functions and solved them using extrema optimization [27], genetic algorithms [1, 28], simulated annealing [29, 35], *expectation-maximization* (EM) [30] and convex optimization [32]. Detection of communities in a complex network is categorized as either non-overlapping or overlapping in terms of node, i.e., vertex, membership value. In non-overlapping community detection, each node belongs to only one community; meanwhile, in overlapping community detection each vertex can belongs to more than one community [36]. Non-overlapping community detection

has attracted a lot of attention [2, 3, 23, 24, 25, 26, 37, 38, 39, 40], and efficient approaches such as those proposed in [3, 40] were developed with respect to performance (modularity) and computational complexity. Raghavan et al. proposed the linear-time method based on *Label Propagation* (LP) [2], which can be considered a new category for non-overlapping community detection, which focuses on large or even massively-sized networks. Some works propose fast overlapping community detection [41, 42]; however, performance of their overlapping community detection is not measured in terms of modularity. Some recent works propose faster techniques for overlapping community detection via modularity maximization. Here, it is aimed to address both overlapping and non overlapping community detection scenario applicable to a wide spectrum of network sizes.

1.1.1 Non-overlapping community detection

Here, a novel LP-based technique is proposed which leads to stable and superior solutions in terms of modularity and computational complexity. Similar to the well-known LP-based techniques, in the proposed algorithm the optimum label for a vertex is selected from labels of its neighbors by maximizing the modularity variation associated with each label transition. Although most LP-based techniques leverage efficient approaches for calculation of modularity gain variation at each label (community membership) transition [3, 40, 43, 44], these methods are still computationally

complex in large networks, where there are hundreds or thousands of candidate labels per node to be evaluated. Evaluation of modularity variation for all available labels dramatically increases computational complexity of the overall community detection procedure. Instead of calculating the actual value of modularity gain corresponding to each label transition, a novel objective function corresponding to all candidate labels is developed. The proposed objective function is simplified into two terms, called the *static* and *dynamic* components. The static component represents the computationally complex term and is calculated via a static label list. However, the dynamic component represents the computationally more-efficient term and is calculated via a dynamic label list. The proposed *Hybrid Label Propagation* (HLP) leverages the pre-calculated values of the static component once per each iteration, while the dynamic component is calculated per each candidate label. This dramatically reduces the overall computational complexity associated with the proposed objective function. The HLP approach produces decent performance in terms of Modularity and *Normalized Mutual Information* (NMI) in near linear-time; however, a generalized version of the proposed objective function called *Modularity Gain Acceleration* (MGA) is introduced to further improve efficiency. Like the HLP approach, MGA divides the modularity gain objective function into two components, called the *Local Sum-Weight* (LSW) and the *General Sum-Weight* (GSW). The LSW is the lower complexity component and is calculated per each label transition, the GSW is more computationally complex

and is calculated only once per each label. Then, the GSW is updated by leveraging a simple process for each node-label transition, rather than direct update for all available labels.

1.1.2 Overlapping community detection

Overlapping community detection is one of the most notable problems in this area. Extensive research has been conducted to develop efficient methods for non-overlapping community detection in large-scale networks, but these approaches are not appropriate for overlapping community detection. In this thesis *Fast Fuzzy Modularity Maximization* (FFMM) for overlapping community detection is studied. FFMM exploits a novel iterative equation for calculation of modularity gain associated with changing the fuzzy membership values of network vertices. The simplicity of the proposed iterative modularity update equation enables efficient modifications, reducing computational complexity to a linear function of the network size $O(N)$, which is advances the current state-of-the-art. In order to apply FFMM to large networks, Multi-Cycle FFMM is proposed. Multi-Cycle FFMM is accomplished in multiple cycles, each using the FFMM approach for community detection. Then, each detected community at each cycle is considered as a sub-network for the next cycle until the desired community resolution is acquired. Simulation results demonstrate that Multi-Cycle FFMM produces a remarkable performance in terms of overlapping modularity

value. We use *Overlapping Normalized Mutual Information* (ONMI) as an evaluation metric to measure quality of detected communities in large-scale networks with over 10^6 nodes.

1.2 Range-Free Anchor Selection in Wireless Sensor Networks Using Community Detection

WSNs have a variety of applications, such as environmental monitoring [45], road traffic monitoring [46, 47], Health [11, 48, 49], etc. For a survey of WSN applications, see [50, 51]. Self-localization capability is highly desirable for most of the mentioned applications. In other words, most WSN applications are location-aware, meaning the measured data or observed events are meaningless without information about the locations where the data are obtained or measured. Thus, the development of approaches to localize sensor nodes in WSNs is a critical research area. Localization in WSNs is categorized as range-free [52, 53] or range-based techniques [54, 55]. Range-free techniques exploit network information such as connectivity while range-based techniques use a variety of information such as Received Signal Strength Indicator (RSSI) [56], Time-of-Arrival (ToA) [57], Time Difference of Arrival (TDoA) [58], Angle of Arrival (AoA) [59] or combinations thereof [14]. Range-based approaches offer higher localization accuracy; however, they impose difficulty in producing range

measurements, such as higher energy consumption. Energy efficiency is very vital in WSNs. Consider that up to 80% of the consumed energy in WSNs is due to radio communication including ranging [60]. Therefore, using a technique which reduces ranging process is promising in WSNs terminology.

1.2.1 Distributed localization in WSNs

Here, it is aimed to develop a criteria that maintains the optimum distribution of localization and ranging within the entire network. The goal is to optimize the range based distributed localization in terms of trade-off between performance and complexity. Considering range based localization, it is aimed to develop the optimum anchor selection approach and study its impact on distributed localization in dense WSNs. To this end, it is vital to study modern distributed localization approaches to exploit the state-of-the-art methods.

Considering a WSN consisting of N sensor nodes and M anchor nodes within a D dimensional space, the objective function that maximizes the likelihood of measured ranges is represented by [61, 62]

$$[\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, ..., \hat{\mathbf{x}}_N] = \underset{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_N}{\mathbf{arg\ min}} \left\{ \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N+M} \frac{1}{\sigma_{ij}^2} \left| d_{i,j}^2 - \|\mathbf{x}_j - \mathbf{x}_i\|^2 \right| \right\}, \tag{1.1}$$

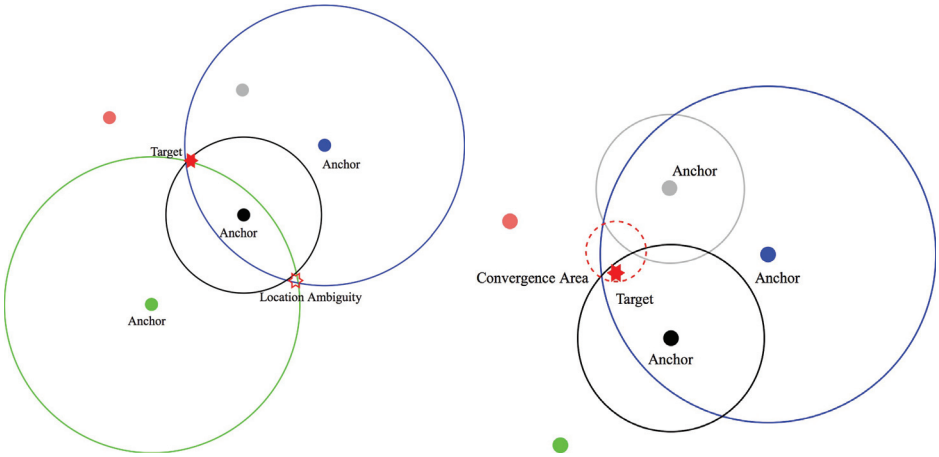
where $\mathbf{x}_i = \left[x_i^{(1)}, x_i^{(1)}, ..., x_i^{(D)}\right]$, $\mathbf{x}_j = \left[x_j^{(1)}, x_j^{(1)}, ..., x_j^{(D)}\right]$ and $d_{i,j}$ denote the coordinates of the ith sensor node and the jth sensor/anchor node, and the measured range between them, respectively. Moreover, σ_{ij}^2 represents the variance of range measurements corresponding to the ith sensor node and the jth anchor node. However, for distributed approaches, each sensor node aims to maximizes the likelihood of its measured ranges corresponding to its selected anchor nodes represented by

$$\hat{\mathbf{x}_i} = \underset{\mathbf{x}_i}{\mathbf{arg\,min}} \left\{ \sum_{j \in \mathcal{N}_i} \frac{1}{\sigma_{ij}^2} \left| d_{i,j}^2 - \|\mathbf{x}_j - \mathbf{x}_i\|^2 \right| \right\}, \tag{1.2}$$

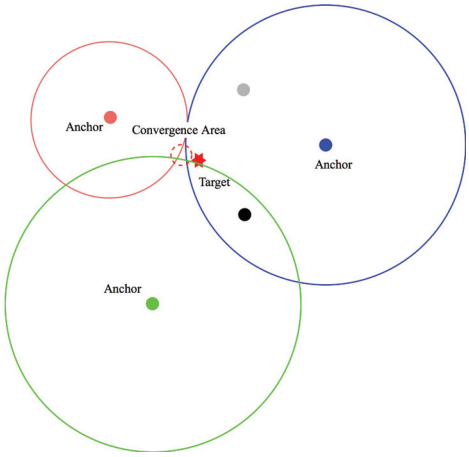
where $\mathcal{N}_i$ represents the list of selected anchor nodes of the ith node. Although using all possible measurements may lead to precise localization, that demands for ranging among sensor and anchor nodes which is not efficient due to high energy consumption associated with ranging techniques [63]. In this study we develop an approach to select the optimum set of anchor nodes using both overlapping and non-overlapping community memberships.

1.2.2 Community detection based range-free anchor selection

Figure 1.1 depicts an example corresponding to a distributed localization scenario. Here, the target node has the advantage of selecting multiple combinations of anchor



(a) Location ambiguity (b) large convergence area



(c) Optimum selection

Figure 1.1: Impact of anchor selection on localization performance, (a) Location ambiguity due to aligned anchors, (b) Large convergence area due to close anchors, (c) Optimum selection

nodes in its vicinity for ranging. The lack of location knowledge, however, hampers the optimum anchor node selection. As shown, non-supervised or random anchor selection may lead to poor anchor node selection, where location ambiguity—shown in Figure 1.1 (a)—or large location convergence area—in Figure 1.1 (b)—is possible. However, by using a supervised approach, it is possible to select the optimum anchor set with minimum location convergence area.

In this work it is aimed to develop a novel technique for anchor selection in very dense WSNs. An optimum anchor selection impacts overall sensor nodes localization processes in terms of accuracy and consumed energy. In this study, we propose using both overlapping and non-overlapping community detection approaches to develop a range-free approach for optimum anchor selection. To this end the network is analyzed within the processing center to reveal both overlapping and/or non-overlapping community memberships. The achieved community membership then will be used in the corresponding objective function to reveal the optimum set of anchors for selected nodes within the network (i.e., nodes that have at least three anchors in the network).

1.3 Dissertation Outline and Contributions

The following chapters summarize my work on the community detection problem along with application-specific contributions to anchor selection based localization in dense *Wireless Sensor Networks* (WSNs). The remainder of this section describes each chapter more concretely and explains the novel contributions of each chapter.

Chapter 2 studies the *Alternating Direction Augmented Lagrangian* (ADAL) method for maximizing a generalized form of Newman's modularity function. First, Newman's modularity is transformed into a quadratic program and then *Completely*

Positive Programming (CPP) is utilized to map the quadratic program to a linear program. This provides the globally optimal maximum modularity cover matrix. In order to solve the proposed CPP problem, a closed form solution using the ADAL approach is proposed.

Chapter 3 studies the novel *Hybrid Label Propagation* (HLP) approach for maximizing a generalized form of Newman's modularity function. Here, a novel objective function is developed to maximize the modularity variation corresponding to each label propagation. Moreover, a hybrid form of synchronous and asynchronous label propagation is developed by using dynamic and static label lists.

Chapter 4 studies Modularity Gain Acceleration (MGA). MGA is a modified version of the proposed objective function which is utilized to reduce complexity of existing label propagation based approaches for community detection. The proposed approach is extremely efficient for very large networks due to its near linear time computational complexity.

Chapter 5 studies *Fast Fuzzy Modularity Maximization* (FFMM) for community detection, which uses a novel iterative equation for calculation of modularity gain associated with changing the fuzzy membership values of network vertices. The simplicity of the proposed iterative modularity update equation enables efficient modifications,

reducing computational complexity to a linear function of the network size $O(N)$, which is beyond the current state-of-the-art. To apply the proposed FFMM to large networks, multi-cycle FFMM is proposed where every detected community at each cycle is considered as an individual sub-network for the next cycle.

Chapter 6 describes the proposed approach for range-free anchor selection based on the proposed overlapping (FFMM) and non-overlapping (HLP) community detection methods. In this approach, sensor nodes select the optimum anchor nodes among available candidates based on associated community structure. Here, two novel objective functions, based on overlapping and non-overlapping community membership, are developed. Simulations over multiple networks with different structures show that the proposed approach improves the average localization error, especially at lower range measurement errors. Finally, **Chapter 7** concludes the dissertation and discusses future works.

Chapter 2

Modularity Maximization Using Completely Positive Programming

2.1 Introduction

Social network analysis has recently attracted a lot of attention. Online users in social networks such as Facebook and Twitter provide a veritable treasure trove of social network data, facilitating significant applications, such as product recommendation systems for on-line retail sites, political election prediction based on discussions of certain topics on Twitter, and so on.

Network analysis is applied over varieties of research topics such as social networks, transportation, anthropology, biology, economics, sociology and bibliometrics studies [5, 6, 7]. Features and characteristics in complex networks provide information associated with a network feature called *community structure*. A community in a network is often defined as a group of nodes—i.e., users—that have more concentrated connections or data in common among the group's members than with the rest of the network. Naturally, nodes with similar attributes will be more likely to form a community. Community detection is one of the most important problems in network analysis. The process in which complex network data are analyzed in order to uncover organizational principles, properties, and structure of complex networks,

The material in this chapter was previously published in Physica A: Statistical Mechanics and its Applications Volume 471, 1 April 2017, Pages 20-32

and ultimately to enable extraction of useful information from them is called community detection. Community detection has attracted much attention in the past two decades. Community detection approaches can be divided into three main categories [17]: traditional methods [18, 19], divisive algorithms [20, 21, 22], and modularity based methods [1, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33]. Traditional methods, such as graph partitioning and clustering, divide or merge clusters based on the similarity between nodes, whereas divisive algorithms are based on removing edges that connect nodes with lower similarity. Modularity based methods work by optimizing an objective function such as *Modularity* introduced by Newman and Girvan [34]. Numerous approaches have been proposed for community detection. Fortunato [17] divided them into three different categories: traditional methods [18, 19], divisive algorithms [20, 21], and modularity based methods [64]-[33]. Traditional methods, such as graph partitioning and clustering, divide or merge clusters based on the similarity between nodes, whereas divisive algorithms are based on removing edges connecting nodes with low similarity. Modularity based methods work by optimizing an objective function such as the modularity introduced by Newman and Girvan [34]. Most of the recent works on community detection are based on modularity maximization proposed by Newman [23]-[31]. Other works have proposed other objective functions and solved them using extremal optimization [37], genetic algorithm [1, 28], simulated annealing [29], *expectation-maximization* (EM) [30] and convex optimization [32].

Finding the partition with maximum modularity is very difficult—NP-hard [67]—due to its non-convexity, and usually yields a sub-optimal partition, e.g., see Fast Unfolding Algorithm [28, 68]. Some approaches, such as spectral optimization [19], greedy methods [23, 65], and [69], extremal optimization [37], and simulated annealing [29] have used searching to obtain solutions for crisp entries of the cover matrix. Although some approaches such as greedy methods, extremal optimization, simulated annealing, and spectral optimization have been used searching the global solution, but the proposed result is crisp. In most of social networks, many of users do not belong to a specific community which cause overlap among communities. To deal with this problem, the crisp overlapping and fuzzy overlapping community structures were proposed in [70]. Crisp overlapping communities let a node belong to more than one community; however, its membership still is binary. In fuzzy overlapping communities, memberships in communities are on the interval $[0, 1]$, and the sum of the memberships for each node is 1. Several works have addressed fuzzy community detection [1, 31] and [33].

In this chapter, we present a novel model for reformulating the crisp modularity maximization problem. First, the objective function is converted to a linear programming problem with completely positive and rank-1 constraints. Then *Alternating Direction Augmented Lagrangian* (ADAL) is applied to solve the *Completely Positive Programming* (CPP) problem, followed by a rank minimization procedure to impose the rank-1

constraint on the final crisp solution. This contribution not only results in a highly-effective algorithm for modularity maximization, but also offers new insight on how to effectively solve the CPP problem with minimum rank. Burer [71] proposed reforming the standard quadratic problem with positive constraints to the linear programming with CPP constraints. However, it does not apply the rank-1 constraint which reconstructs the desired solution (the variable vector in the quadratic problem) from the final solution (the variable matrix in the CPP problem). To the best of our knowledge, this work is the first work to address the modularity maximization problem by reforming its quadratic form to the linear programming problem. However, the proposed approach for solving the quadratic problem with positive constraints can be applied to any other standard problem. The main contributions of this chapter can be summarized as follows:

1. Reformulation of the modularity maximization problem to a linear program with completely positive and rank-1 constraints;

2. Use of the ADAL method to solve the CPP problem;

3. Application of rank minimization algorithm to the ADAL method to minimize the rank of the ADAL output without contravening its optimality;

4. Application of the proposed method to several benchmark networks to investigate the optimality of the obtained cover matrices;

5. Discussion on the limitations and scalability of the proposed algorithm for large scale networks.

The rest of this chapter is organized as follows. Section 2.2 introduces the mathematical model of the community detection problem. The proposed algorithm for community detection is presented in Section 2.3. Section 2.4 presents experiments, analysis, and discussions. Table2.1 contains a selected list of notations and symbols used here.

2.2 Problem Definition

2.2.1 Generalized modularity function

Every social network can be represented by a graph $G = (V, E, \mathbf{W})$, where V is a set of n vertices, E is a set of edges, and $\mathbf{W}$ is an $n \times n$ edge weight(or adjacency) matrix, where w_{ij} in $\mathbf{W}$ denotes the weight of the edge connecting node i and node j. Community detection for a network is the process of finding a $c \times n$ partition matrix (or in graph theory, a cover matrix) $\mathbf{U}$, where each element u_{ki} in $\mathbf{U}$, $k = [c]$; $i = [n]$, is the membership of the ith node in the kth community. There are three main types

Table 2.1
Symbols and corresponding descriptions

Symbol	Description
n	Number of nodes (vertices) in network
c	Number of communities
G	Graph $G = (V; E; \mathbf{W})$
$\mathbf{b}$	Vector of ones
$\mathbb{R}^n$	The n-dimensional Euclidean space
$\mathbb{R}^n_+$	The nonnegative orthant of $\mathbb{R}^n$
$\mathbb{R}^{n\times n}$	The set of real, $n \times n$ matrices
$\mathbb{S}^n$	The set of symmetric matrices in $\mathbb{R}^{n\times n}$
$\mathbb{S}^n_+$	The set of positive semidefinite symmetric matrices (SDP cone)
$\mathbb{S}^{n*}_+$	The dual of SDP cone
$\mathbb{C}$	The cone of completely positive matrices, ($\mathbb{B} \in \mathbb{S}^n : x^T\mathbb{B}x \geq 0$ for all $x \in \mathbb{R}^{n+}$)
$\mathbb{C}^*$	The dual of completely positive cone $\mathbb{C}$
$\mathbb{P}$	The symmetric positive cone (the cone of $n \times n$ real symmetric matrices with non-negative elements)
$\mathbb{P}^*$	The dual of the symmetric positive cone $\mathbb{P}$
$[t]$	The set of integers from 1 to t
$\mathbf{W}$	The adjacency matrix,$\mathbf{W} \in \mathbb{R}^{n\times n}$
$\mathbf{m}$	Degree vector $\mathbf{m} = (\mathbf{m}_1, \ldots, \mathbf{m}_n)^T$
B	Modularity matrix $\mathbf{B} = \mathbf{W} - (\mathbf{m} * \mathbf{m}^T)/\|\mathbf{W}\|$
$\mathbf{U}$	Partition or cover matrix, $\mathbf{U} = [u_{ij}]^{c\times n}$,$u_{ij} \in [0, 1]$
m_i	Degree of vertex v_i
u_i	ith column of $\mathbf{U}$
$\|\mathbf{x}\|$	The norm of a vector $\mathbf{x} \in \mathbb{R}^n$ is denoted by $\sqrt{\mathbf{x}^T\mathbf{x}}$
$tr(.)$	Denotes the sum of the diagonal entries of a matrix
$\langle \mathbf{M}, \mathbf{N} \rangle$	The inner product of $\mathbf{M}$ and $\mathbf{N}$ or $trace(\mathbf{M}^T\mathbf{N})$.
$\mathbf{M} \otimes \mathbf{N}$	The Kronecker product
diag(M)	Vector with the diagonal of the matrix M as its entries
diag($\mathbf{y}$)	Diagonal matrix with elements of the vector $\mathbf{y}$ as its entries
$\mathbf{M} \succeq 0$	$\mathbf{M} \in$ SDP cone($\mathbf{M}$ is positive semidefinite)
$\mathcal{A}(\cdot)$	The linear mapping from the symmetric cone $\mathbb{S}^{c\times n}$ to $\mathbb{R}^{n\times 1}$
$\mathcal{A}^*(\cdot)$	The the adjoint operator of $\mathcal{A}(\cdot)$

of partitions [72]:

$$M_{pc\times n} = \Big\{ \mathbf{U} \in R^{c\times n}; 0 \leq u_{ki} \leq 1, \forall\ k, i; \sum_{k=1}^{c} u_{ki} \leq c, \forall\ i; \sum_{i=1}^{n} u_{ki} < n, \forall\ k \Big\}; \tag{2.1a}$$

$$M_{fc\times n} = \left\{ \mathbf{U} \in M_{pc\times n}; \sum_{k=1}^{c} u_{ki} = 1, \forall\ i \right\}; \tag{2.1b}$$

$$M_{hc\times n} = \{ \mathbf{U} \in M_{fc\times n}; u_{ki} \in \{0, 1\} \}; \tag{2.1c}$$

where $M_{pc\times n}$ is the set of probabilistic; $M_{fc\times n}$ is the set of fuzzy, and $M_{hc\times n}$ is the set of crisp partitions. Much work has been done on crisp community detection, i.e., searching for the best $\mathbf{U} \in M_{hc\times n}$. Other works have focused on fuzzy community detection, i.e., searching for the best $\mathbf{U} \in M_{fc\times n}$. In this work, we propose a generalized community detection algorithm for finding partitions in $M_{hc\times n}$, deriving the crisp partition by hardening with the maximum membership rule. Recently, modularity based methods have been very popular among social network researchers. For the community detection problem, modularity works as the objective function to evaluate the goodness of a given community represented by a partition $\mathbf{U}$. Modularity was originally introduced by Newman and Girvan [73] as a way to evaluate crisp communities in networks. It is defined as

$$\mathcal{Q} = \frac{1}{\|\mathbf{W}\|} \sum_{i=1,j=1}^{n} \left(w_{ij} - \frac{m_i m_j}{\|\mathbf{W}\|} \right) \delta(i,j), \tag{2.2}$$

where $m_i = \sum_{j=1}^{n} w_{ij}, i = [n], \|\mathbf{W}\| = \sum_{i=1}^{n} m_i$ and $\delta(i,j) = 1$ if node i and node j are in the same community; else $\delta(i,j) = 0$. Liu [29] proposed a modified modularity and combined it with a simulated annealing approach for fuzzy community detection. Later, Havens et al. [36] introduced a more generalized modularity, given at (2.3), that works for evaluating not only crisp partitions, but also fuzzy and possibilistic partitions:

$$\mathcal{Q}_g = \frac{\operatorname{tr}\left(\mathbf{UBU^T}\right)}{\|\mathbf{W}\|}, \tag{2.3}$$

where $\mathbf{B} = \left[\mathbf{W} - \frac{\mathbf{m}^T\mathbf{m}}{\|\mathbf{W}\|}\right]$ and $\mathbf{m} = (\mathbf{m_1}, \mathbf{m_2}, \ldots, \mathbf{m_n})^T$. Considering the generalized modularity at (2.3) as the objective function for community detection, we reform it as a standard quadratic function in the next subsection.

2.2.2 Objective function

Obtaining the extremum of the modified modularity function given at (2.3) is equivalent to finding the extremum of the numerator, $\text{tr}\left(\mathbf{UBU}^T\right)$, as the denominator is constant with $\mathbf{U}$. Let $\mathbf{x} = vec(\mathbf{U}^T)$ and $\mathbf{Q} = \mathbf{I} \otimes \mathbf{B}$, where $\mathbf{I}$ refers to the identity matrix and $\mathbf{B}$ is the $n \times n$ modularity matrix. The term $\text{tr}(\mathbf{UBU^T})$ can be formulated as the standard quadratic form of $\mathbf{x}^T\mathbf{Q}\mathbf{x}$ where are $\mathbf{x}$ is a $(c \times n) \times 1$ vector and $\mathbf{Q}$ is a $(c \times n) \times (c \times n)$ symmetric matrix. Applying $\mathbf{x}^T\mathbf{Q}\mathbf{x} = trace(\mathbf{Q} \times (\mathbf{x}\mathbf{x}^T))$, leads to the standard linear programming objective function,

$$\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X}, \tag{2.4}$$

where $\mathbf{X} := \mathbf{x}\mathbf{x}^T$ and $\mathbf{C} = -\mathbf{Q}$. We must also apply the summation constraint on $\mathbf{U}$, i.e., $\sum_{i=1}^{c} u_{ij} = 1, i = [n]$. Considering $\mathbf{x} = vec(\mathbf{U}^T)$ gives $\sum_{i=1}^{c} u_{ij} = \sum_{i=1}^{c} x_{(i-1)\times n+j}$, which can be represented as the matrix form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{b}$ denotes a vector of ones by size $n \times 1$ and matrix $\mathbf{A}$ is

$$\mathbf{A}_{ij} = \mathbf{b} \otimes \mathbf{I} = \begin{cases} 1, & i = [n], j = n \times (k-1) + i, k = [c]; \\ 0, & else. \end{cases} \tag{2.5}$$

However, any constraint on (2.4) needs to be defined on the elements of $\mathbf{X}$. Multiplying each side of $\mathbf{Ax} = \mathbf{b}$ by its transpose, and taking the diagonal leads to $\mathbf{Ax} = \mathbf{b}$ and $diag\left\{\mathbf{AXA}^T\right\} = \mathbf{b}^2$ [74], which gives the new form of (2.4) as

$$\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X}, \text{ s.t. } diag\left\{\mathbf{AXA}^T\right\} = \mathbf{b}^2. \tag{2.6}$$

The last constraint that needs to be applied is $0 \leq u_{ij} \leq 1$, $\forall i, j$. Considering $\mathbf{X} := \mathbf{x}\mathbf{x}^T$ and $0 \leq u_{ij} \leq 1$ reveals two significant properties on the desired final solution for $\mathbf{X}$. First, $\mathbf{X}$ is a *completely positive* (CP) matrix, as it can be decomposed to a vector that has only positive elements. Second, the rank of $\mathbf{X}$ is 1. Therefore, by applying these two constraints, the final version of our proposed modularity maximization objective function for crisp community detection is

$$\begin{aligned} &\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X}, \text{ s.t. } diag\left\{\mathbf{AXA}^T\right\} = \mathbf{b}^2, \\ &\mathbf{X} \in \text{Completely Positive Cone}(\mathbb{C}),\ rank(\mathbf{X}) = 1. \end{aligned} \tag{2.7}$$

The defined problem at (2.7) is quite similar to the standard format of *semi-positive definite* (SDP) problems [75]. Several types of solvers such as SeDuMi [75] can be utilized for solving such programming problems. However, our objective function at

(2.7) has the CP constraint which is a much tighter constraint than SDP; therefore, none of the available solvers can be applied. This motivates our proposed modularity maximization method for community detection in networks, discussed in the next section.

2.3 Proposed Method

In this section we apply the *Alternative Direction Augmented Lagrangian* (ADAL) method to develop an iterative algorithm for finding the maximum modularity partition matrix. We first describe how to apply ADAL for solving the standard SDP in Section 2.3.1, then we show how to add the CP constraint to the standard SDP problem in Section 2.3.2. Section 2.3.3 describes our procedure for imposing the rank 1 constraint on the solution, $\mathbf{X}$. The full algorithm is summarized in 2.3.4.

2.3.1 ADAL for standard SDP programming

In this section the standard SDP programming problem is introduced. We first derive the dual problem from the primal and then derive the *Karush-Kuhn-Tucker* (KKT) conditions. We then show how to apply the ADAL method for solving the SDP dual problem.

The standard form of the SDP problem has the primal form

$$\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X}, \text{ s.t. } \mathcal{A}(\mathbf{X}) = \mathbf{b}, \ \mathbf{X} \succeq 0, \tag{2.8}$$

where $\mathbf{X} \succeq 0$ indicates that $\mathbf{X} \in$ SDP Cone$(\mathbb{S}_+^n)$ and $\mathcal{A}(\mathbf{X}) : \mathbb{S}^{c\times n} \to \mathbb{R}^{n\times 1}$ represents the linear mapping from the symmetric cone to $\mathbb{R}^{n\times 1}$,

$$\mathcal{A}(\mathbf{X}) = diag(\mathbf{A}\mathbf{X}\mathbf{A}^T) = \mathbf{b}^2. \tag{2.9}$$

The adjoint operator of $\mathcal{A}(\mathbf{X})$ is $\mathcal{A}^*(\mathbf{y}) : \mathbb{R}^{n\times 1} \to \mathbb{S}^{c\times n}$, such that (see Proposition1 in Appendix A)

$$\mathcal{A}^*(\mathbf{y}) = \mathbf{A}^T diag(\mathbf{y})\mathbf{A}. \tag{2.10}$$

The Lagrangian function of the primal problem at (2.8) is [76]

$$L(\mathbf{X}, \lambda) = \langle \mathbf{C}, \mathbf{X} \rangle + \langle \lambda, \mathcal{A}(\mathbf{X}) - \mathbf{b} \rangle , \tag{2.11}$$

where $\lambda \in \mathbb{R}^{n\times 1}$ are the *Lagrange multipliers*. Applying an algebraic manipulation to (2.11) leads to

$$\begin{aligned} L(\mathbf{X}, \lambda) &= \langle \mathbf{C}, \mathbf{X} \rangle + \langle \lambda, \mathcal{A}(\mathbf{X}) \rangle + \langle \lambda, -\mathbf{b} \rangle , \\ &= \langle \mathbf{C}, \mathbf{X} \rangle + \langle \mathcal{A}^*(\lambda), \mathbf{X} \rangle + \langle \lambda, -\mathbf{b} \rangle , \\ &= \langle \mathbf{C} + \mathcal{A}^*(\lambda), \mathbf{X} \rangle + \langle \lambda, -\mathbf{b} \rangle . \end{aligned} \tag{2.12}$$

It can be observed that every feasible solution $\mathbf{X}^*$, such that $\mathcal{A}(\mathbf{X}^*) = \mathbf{b}$ and $L(\mathbf{X}^*, \mathbf{y}) = \langle \mathbf{C}, \mathbf{X}^* \rangle$, can be considered as the Lagrange multipliers corresponding to the feasible solution of the dual problem that has the same Lagrangian function (subject to $\mathbf{X}$ as Lagrange multipliers). Given $\mathbf{y} := (-\lambda)$, (2.12) can be considered as the Lagrangian function of the following dual problem,

$$\max_{\mathbf{y}} \mathbf{b}^T \mathbf{y}, \text{ s.t. } \mathbf{C} - \mathcal{A}^*(\mathbf{y}) = \mathbf{S}, \ \mathbf{S} \in \mathbb{S}_+^{n*}, \tag{2.13}$$

where $\mathcal{A}^*(.)$ is the adjoint operator of $\mathcal{A}(\cdot)$ at (2.10) and $\mathbb{S}_+^{n*}$ is the dual of the SDP cone ($\mathbb{S}_+^n$). However, the SDP cone is self-adjoint, i.e., $\mathbb{S}_+^n = \mathbb{S}_+^{n*}$. Therefore, the dual problem can be revised to

$$\max_{\mathbf{y}} \mathbf{b}^T \mathbf{y}, \text{ s.t. } \mathcal{A}^*(\mathbf{y}) + \mathbf{S} = \mathbf{C}, \ \mathbf{S} \succeq 0. \tag{2.14}$$

The duality gap of the primal and dual problems is

$$\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X} - \max_{\mathbf{y}} \mathbf{b}^T \mathbf{y} = \min_{\mathbf{X}} \langle \mathbf{X}, \mathbf{C} \rangle - \max_{\mathbf{y}} \langle \mathbf{b}, \mathbf{y} \rangle , \tag{2.15}$$

satisfying $\mathcal{A}(\mathbf{X}) = \mathbf{b}$ and $\mathcal{A}^*(\mathbf{y}) + \mathbf{S} = \mathbf{C}$. This leads to

$$\begin{aligned} &\min_{\mathbf{X}} \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) + \mathbf{S} \rangle - \max_{\mathbf{y}} \langle \mathcal{A}(\mathbf{X}), \mathbf{y} \rangle \\ &= \min_{\mathbf{X}} \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) + \mathbf{S} \rangle - \max_{\mathbf{y}} \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) \rangle . \end{aligned} \tag{2.16}$$

It can be observed that $\min_{\mathbf{X}} \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) + \mathbf{S} \rangle = \max_{\mathbf{y}} \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) \rangle$ when $\langle \mathbf{X}, \mathbf{S} \rangle = 0$. Therefore, the KKT conditions for the primal and dual problems are

$$\begin{cases} \mathcal{A}(\mathbf{X}) = \mathbf{b}, \mathbf{X} \in \mathbb{S}_+^n; \\ \mathcal{A}^*(\mathbf{y}) + \mathbf{S} = \mathbf{C}, \mathbf{S} \in \mathbb{S}_+^{n*}; \\ \langle \mathbf{X}, \mathbf{S} \rangle = 0. \end{cases} \tag{2.17}$$

These KKT conditions guarantee that if both problems (the primal and the dual) are strictly feasible—$\mathcal{A}(\mathbf{X}) = \mathbf{b}$ and $\mathcal{A}^*(\mathbf{y}) + \mathbf{S} = \mathbf{C}$ can be satisfied—then the duality gap is zero and the dual problem can attain the optimal solution. Since finding the optimum solution for (2.14) is much simpler, usually the dual problem is solved. The Lagrangian function of proposed the dual problem defined in Equation (2.14) is [76]

$$\begin{aligned} L(\mathbf{y}, \mathbf{S}, \mathbf{X}) = -\mathbf{b}^T\mathbf{y} + \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) + \mathbf{S} - \mathbf{C} \rangle \\ + 1/(2\mu) \left\| \mathcal{A}^*(\mathbf{y}) + \mathbf{S} - \mathbf{C} \right\|^2. \end{aligned} \tag{2.18}$$

The proposed Lagrangian function at (2.18) has the standard format which applies the constraints as penalty functions within the objective function. However, forming the Lagrangian function is not the problem here; the main issue is how to solve for $\mathbf{y}$, $\mathbf{X}$, and $\mathbf{S}$, such that $\mathbf{S} \succeq 0$. Starting from $\mathbf{X} = [\mathbf{0}]$, the augmented Lagrangian method solves $L(\mathbf{y}, \mathbf{S}, \mathbf{X})$ on the kth iteration for $\mathbf{y}^{k+1}$ and $\mathbf{S}^{k+1}$, and then updates

the Lagrangian multipliers $\mathbf{X}^{k+1}$ by applying [76]

$$\mathbf{X}^{k+1} = \mathbf{X}^{k} + \frac{\mathcal{A}^{*}(\mathbf{y}^{k+1}) + \mathbf{S}^{k+1} - \mathbf{C}}{\mu}. \tag{2.19}$$

Finding the optimum solution of $L(\mathbf{y}, \mathbf{S}, \mathbf{X})$ as a function of $\mathbf{y}$ and $\mathbf{S}$ is the only remaining issue. The authors of [76] simplified this problem by minimizing $L(\mathbf{y}, \mathbf{S}, \mathbf{X})$ for $\mathbf{y}$ (given a fixed $\mathbf{S}$) and then minimized $L(\mathbf{y}, \mathbf{S}, \mathbf{X})$ for $\mathbf{S}$ (considering $\mathbf{S} \succeq 0$ and fixed $\mathbf{y}$) such that $\mathbf{y}^{*} : \nabla_{y} L(\mathbf{y}^{*}, \mathbf{S}, \mathbf{X}) = 0$. This leads to (see Proposition 2 in the Appendix A)

$$\mathbf{y}^{*} = \left(\mathcal{A}\mathcal{A}^{*}\right)^{-1} \left[\left(\mathbf{b}^{T} - \mathcal{A}\left(\mathbf{X}\right)\right)\mu - \mathcal{A}\left(\mathbf{S} - \mathbf{C}\right)\right]. \tag{2.20}$$

Here, $\mathcal{A}\mathcal{A}^{*}$ is an invertible matrix such that $\mathcal{A}\left(\mathcal{A}^{*}\left(\mathbf{y}\right)\right) = \left(\mathcal{A}\mathcal{A}^{*}\right)\mathbf{y}$. Considering (2.5), it can be shown that $\mathbf{A}\mathbf{A}^{T} = c\mathbf{I}$ where $\mathbf{I}$ is the identity matrix. This leads to the closed form solution,

$$\begin{aligned}
\mathcal{A}\left(\mathcal{A}^{*}\left(\mathbf{y}\right)\right) &= \mathcal{A}\left(\mathbf{A}^{T} diag\left\{\mathbf{y}\right\}\mathbf{A}\right), \\
&= diag\left\{\mathbf{A}\left(\mathbf{A}^{T} diag\left\{\mathbf{y}\right\}\mathbf{A}\right)\mathbf{A}^{T}\right\}, \\
&= diag\left\{\mathbf{A}\mathbf{A}^{T} diag\left\{\mathbf{y}\right\}\mathbf{A}\mathbf{A}^{T}\right\}, \\
&= c\mathbf{I}\left[diag\left\{\mathbf{y}\right\}\right]\mathbf{I}c = c^{2}\mathbf{I}\mathbf{y}.
\end{aligned} \tag{2.21}$$

This indicates that for our modularity maximization problem, $\mathcal{A}\mathcal{A}^{*} = c^{2}\mathbf{I}$ and $\left(\mathcal{A}\mathcal{A}^{*}\right)^{-1} = \frac{1}{c^{2}}\mathbf{I}$.

To find the optimum $\mathbf{S}$, the Lagrangian function, $L(\mathbf{y}, \mathbf{S}, \mathbf{X})$ must be minimized as a function of $\mathbf{S}$, fixing $\mathbf{y}$. However, the SDP constraint on $\mathbf{S}$ hinders the application of the same procedure used for obtaining the optimal $\mathbf{y}$. Consider the Lagrangian function,

$$L(\mathbf{S}|\mathbf{y}, \mathbf{X}) = \langle \mathbf{X}, \mathbf{S} \rangle + 1/(2\mu) \left\| \mathcal{A}^*(\mathbf{y}) + \mathbf{S} - \mathbf{C} \right\|^2. \tag{2.22}$$

The optimum solution for S is obtained by the optimization (see Proposition 3 in the Appendix for proof),

$$\mathbf{S}^* = \min_{\mathbf{S}} \left\{ \left\| \mathbf{S}^{(k)} - \mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{X}^{(k)}\right) \right\|_F^2 \right\}, \ \mathbf{S} \succeq 0. \tag{2.23}$$

The optimum solution of (2.23) is obtained by projecting $\mathbf{V}$ onto the SDP cone ($\mathbb{S}_+^{n*}$), and is achieved by spectral decomposition of $\mathbf{V}$. First, $\mathbf{V}$ is decomposed into its eigenvectors with positive and negative eigenvalues,

$$\begin{aligned} \mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{X}^{(k)}\right) &= \mathbf{Q}\mathbf{\Sigma}\mathbf{Q}^*, \\ &= [\mathbf{Q}_+ \ \mathbf{Q}_-] \begin{bmatrix} \mathbf{\Sigma}_+ & 0 \\ 0 & \mathbf{\Sigma}_- \end{bmatrix} \begin{bmatrix} \mathbf{Q}_+^* \\ \mathbf{Q}_-^* \end{bmatrix}. \end{aligned} \tag{2.24}$$

where $\{\mathbf{\Sigma}_+, \mathbf{Q}_+\}$ denotes the positive eigenvalues and their corresponding eigenvectors, and $\{\mathbf{\Sigma}_-, \mathbf{Q}_-\}$ denotes the negative eigenvalues and their corresponding eigenvectors of matrix $\mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{X}^{(k)}\right)$. This implies that $\mathbf{S}^{(k+1)}$ can be achieved after decomposition of $\mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{X}^{(k)}\right)$ by applying

$$\mathbf{S}^{(k+1)} = \mathbf{Q}_+ \mathbf{\Sigma}_+ \mathbf{Q}_+^*. \tag{2.25}$$

The final solution for the proposed dual problem at (2.14) is outlined in Algorithm 1.

2.3.2 Adding the positivity constraint

The proposed solution in Algorithm 1 is for an SDP cone; however, we expect a matrix within the CP cone ($\mathbb{C}$). The CP cone is more limited as compared to the SDP cone; every matrix in the CP cone is in the SDP cone; however, the opposite is not true. To achieve a solution to our problem at (2.7), the positively constraint on

Algorithm 1: Alternative Direction Augmented Lagrangian for Semi-Positive Definite Programming

Require: $\mathbf{C}, \mathbf{A}, \mathbf{b}$

return $\mathbf{X}$

Initialize $\mu, \mathbf{S}^{(0)}$ and $\mathbf{X}^{(0)}$.

while $|\mathbf{y}^{k+1} - \mathbf{y}^k| < \epsilon_y$ **do**

$\mathbf{y}^{(k+1)} = (\mathcal{A}\mathcal{A}^*)^{-1}\left[\left(\mathbf{b}^T - \mathcal{A}\left(\mathbf{X}^{(k)}\right)\right)\mu - \mathcal{A}\left(\mathbf{S}^{(k)} - \mathbf{C}\right)\right]$.

$\mathbf{S}^{(k+1)} = \underset{\mathbb{S}_+^{n*}}{Proj}(\mathbf{C} - \mathcal{A}^*(\mathbf{y}^{(k+1)}) - \mathbf{X}^{(k)}\mu)$.

$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + (\mathcal{A}^*(\mathbf{y}^{(k+1)}) + \mathbf{S}^{(k+1)} - \mathbf{C})/\mu$

end while

elements of $\mathbf{X}$ must be added to the final format of objective function as

$$\begin{aligned} &\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X}, \text{ s.t. } \mathcal{A}(\mathbf{X}) = \mathbf{b}, \\ &\mathbf{X} \succeq 0, \ \mathbf{X} \in \text{Symmetric Positive Cone}(\mathbb{P}), \\ &\mathbf{X} \in \text{SDP cone}(\mathbb{S}_+^n). \end{aligned} \tag{2.26}$$

Here, the new constraint $\mathbf{X} \in$ Positive Cone($\mathbb{P}$) ensures that the elements of $\mathbf{X}$ are positive. Similar to (2.8) the dual problem of (2.26) is [77]

$$\begin{aligned} &\max_{\mathbf{y}} \mathbf{b}^T\mathbf{y}, \text{ s.t. } \mathcal{A}^*(\mathbf{y}) + \mathbf{S} + \mathbf{Z} = \mathbf{C}, \\ &\mathbf{S} \in \text{Dual of SDP Cone}(\mathbb{S}_+^{n*}), \\ &\mathbf{Z} \in \text{Dual of Symmetric Positive Cone}(\mathbb{P}^*), \end{aligned} \tag{2.27}$$

where $\mathbf{Z}$ is the second slack variable of the dual problem that bounds the positivity on $\mathbf{X}$. The Lagrangian function corresponding to the objective function proposed at (2.26) is [78]

$$\begin{aligned} L(\mathbf{y}, \mathbf{S}, \mathbf{Z}, \mathbf{X}) = &-\mathbf{b}^T\mathbf{y} + \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) + \mathbf{S} + \mathbf{Z} - \mathbf{C} \rangle \\ &+ 1/(2\mu) \left\| \mathcal{A}^*(\mathbf{y}) + \mathbf{S} + \mathbf{Z} - \mathbf{C} \right\|^2. \end{aligned} \tag{2.28}$$

Applying the same procedure optimizing the Lagrangian function as function of $\mathbf{y}$, $\mathbf{S}$, $\mathbf{Z}$ and $\mathbf{X}$, one-by-one leads to the update equation,

$$\mathbf{y}^* = (\mathcal{A}\mathcal{A}^*)^{-1} \left[\left(\mathbf{b}^T - \mathcal{A}(\mathbf{X}) \right) \mu - \mathcal{A}(\mathbf{S} + \mathbf{Z} - \mathbf{C}) \right]. \tag{2.29}$$

The optimum solution of $\mathbf{S}$ and $\mathbf{Z}$ are obtained by minimizing (2.28) with respect to $\mathbf{S}$ and $\mathbf{Z}$. Applying the same procedure as for the SDP cone, the update equations for $\mathbf{S}$ and $\mathbf{Z}$ are

$$\begin{aligned}\mathbf{S}^{(k+1)} &= \min_{\mathbf{S}} \left\{ \left\| \mathbf{S}^{(k)} - \mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{Z}^{(k)}, \mathbf{X}^{(k)}\right) \right\|_F^2 \right\}, \\ &= \underset{\mathbb{S}_+^{n*}}{Proj}\left(\mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{Z}^{(k)}, \mathbf{X}^{(k)}\right)\right), \\ &= \mathbf{Q}_+ \mathbf{\Sigma}_+ \mathbf{Q}_+^* \ ;\end{aligned} \tag{2.30}$$

$$\begin{aligned}\mathbf{Z}^{(k+1)} &= \min_{\mathbf{Z}} \left\{ \left\| \mathbf{Z}^{(k)} - \mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{S}^{(k+1)}, \mathbf{X}^{(k)}\right) \right\|_F^2 \right\}, \\ &= \underset{\mathbb{P}^*}{Proj}\left(\mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{S}^{(k+1)}, \mathbf{X}^{(k)}\right)\right).\end{aligned} \tag{2.31}$$

However, unlike for $\mathbb{S}_+^{n*}$, the $\mathbb{P}^*$ cone is not self-adjoint and needs to be calculated by applying the Moreau decomposition theorem [79], which states that, for any symmetric matrix $\mathbf{R} \in \mathcal{X}$ and for any closed convex cone such that $\mathbb{K} \in \mathcal{X}$,where $\mathcal{X}$ denotes a finite-dimensional Euclidean space such that $\mathbb{S}_+^n \bigcap \mathbb{P}$ is non empty we have

$$\underset{\mathbb{K}^*}{Proj}(\mathbf{R}) = \mathbf{R} + \underset{\mathbb{K}}{Proj}(-\mathbf{R}). \tag{2.32}$$

Algorithm 2: Alternative Direction Augmented Lagrangian for Completely Positive Programming

Require: $\mathbf{C}, \mathbf{A}, \mathbf{b}$

return $\mathbf{X}$

Initialize $\mu, \mathbf{S}^{(0)}, \mathbf{Z}^{(0)}$ and $\mathbf{X}^{(0)}$.

while $|\mathbf{y}^{k+1} - \mathbf{y}^{k}| < \epsilon_y$ **do**

$\mathbf{y}^{(k+1)} = (\mathcal{A}\mathcal{A}^*)^{-1} \left[\left(\mathbf{b}^T - \mathcal{A}\left(\mathbf{X}^{(k)}\right)\right)\mu - \mathcal{A}\left(\mathbf{S}^{(k)} + \mathbf{Z}^{(k)} - \mathbf{C}\right) \right]$.

$\mathbf{S}^{(k+1)} = \underset{\mathbb{S}_+^{n*}}{Proj}(\mathbf{C} - \mathbf{Z}^{(k)} - \mathcal{A}^*(\mathbf{y}^{(k+1)}) - \mathbf{X}^{(k)}\mu)$.

$\mathbf{Z}^{(k+1)} = \underset{\mathbb{P}^*}{Proj}(\mathbf{C} - \mathbf{S}^{(k+1)} - \mathcal{A}^*(\mathbf{y}^{(k+1)}) - \mathbf{X}^{(k)}\mu)$.

$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + (\mathcal{A}^*(\mathbf{y}^{(k+1)}) + \mathbf{S}^{(k+1)} + \mathbf{Z}^{(k+1)} - \mathbf{C})/\mu$.

end while

This leads to the final equation for $\mathbf{Z}$,

$$\begin{aligned} \mathbf{Z}^{(k+1)} &= \underset{\mathbb{P}^*}{Proj}\left(\mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{S}^{(k+1)}, \mathbf{X}^{(k)}\right)\right), \\ &= \mathbf{V}\left(\mathbf{y}^{(k+1)}, \mathbf{S}^{(k+1)}, \mathbf{X}^{(k)}\right) + \underset{\mathbb{P}}{Proj}\left(-V\left(\mathbf{y}^{(k+1)}, \mathbf{S}^{(k+1)}, \mathbf{X}^{(k)}\right)\right). \end{aligned} \tag{2.33}$$

The $\underset{\mathbb{P}}{Proj}(\mathbf{V})$ operator replaces the negative values in $\mathbf{V}$ with zeros. The final solution for the proposed dual problem at (2.27) is outlined in Algorithm 2.

2.3.3 Applying the rank-1 constraint

The last constraint that must be added to the proposed objective function at (2.26) is the *rank-1* constraint on $\mathbf{X}$. This is an extremely important constraint, which allows us to decompose $\mathbf{X}$ as $\mathbf{X} = \mathbf{x}\mathbf{x}^T$, and finally construct the partition matrix $\mathbf{U}$ from $\mathbf{x}$. However, the rank-1 cone is nonconvex and cannot be directly applied to any convex optimization approaches, including the ADAL method. Therefore, no direct method

Algorithm 3: Rank Minimization

Require: $\mathbf{X}$, ζ

1: **return** $\mathbf{X}^{\dagger}$
2: Decompose $\mathbf{X}$ to find its eigenvalue matrix, $\mathbf{V}_X$, and eigenvector matrix, $\mathbf{U}_X$.
3: Set the smallest non-zero eigenvalue within $\mathbf{V}_X$ to zero, forming $\hat{\mathbf{V}}_{\mathbf{X}}$.
4: $\mathbf{X}^{\dagger} = \mathbf{U}_X \hat{\mathbf{V}}_X \mathbf{U}_X^T$.
5: **while** $\frac{||\mathbf{X}-\mathbf{X}^{\dagger}||_F^2}{||\mathbf{X}||_F^2} < \zeta$ **do**
6: Set the smallest non-zero eigenvalue within $\hat{\mathbf{V}}_X$ to zero, forming a new $\hat{\mathbf{V}}_X$.
7: $\mathbf{X}^{\dagger} = \mathbf{U}_X \hat{\mathbf{V}}_X \mathbf{U}_X^T$.
8: **end while**
9: Return the last eigenvalue and form $\hat{\mathbf{V}}_X$.
10: Compute $\mathbf{X}^{\dagger} = \mathbf{U}_X \hat{\mathbf{V}}_X \mathbf{U}_X^T$ as output.

exists to force the solution of the ADAL CPP method into the rank-1 cone.

The approach that is applied in this work is quite simple and effective. Here, the calculated $\mathbf{X}$ at each iteration is passed through a rank minimization function which finds the minimum rank matrix that is close to input $\mathbf{X}$ such that $\frac{||\mathbf{X}-\mathbf{X}^{\dagger}||_F^2}{||\mathbf{X}||_F^2} \leq \zeta$. This can be implemented using Algorithm 3.

The rank minimization algorithm reduces the rank of $\mathbf{X}$ according to a normalized error. At each iteration of steps 4–7 in algorithm 3, the rank of $\mathbf{X}$ is reduced by one. Hence, the parameter ζ determines the acceptable error between the input $\mathbf{X}$ and its rank-reduced output $\mathbf{X}^{\dagger}$. Algorithm 3 is inserted into Algorithm 2 after step 6, reducing the rank of $\mathbf{X}^{(k+1)}$ at each iteration.

2.3.4 Proposed method summary

The proposed ADAL-based method for finding the optimum solution of the proposed CPP problem at (2.26) and (2.27) consists of using Algorithm 2 to optimize the CPP problem and Algorithm 3 for rank minimization. After reaching the stopping criterion in Algorithm 2, the cover matrix $\mathbf{U}$ can be constructed using the inverse operation of $\mathbf{x} = vec(\mathbf{U})$, i.e., the cover matrix $\mathbf{U}$ of size $c \times n$ can be constructed using $\mathbf{x}$, where $\mathbf{x}$ is the corresponding eigenvector of the only non-zero or prominent eigenvalue of the CPP algorithm output, $\mathbf{X}$. Finally, the desired crisp partition matrix $\mathbf{U}^c = [\mathbf{u}_1^c, \mathbf{u}_2^c, ..., \mathbf{u}_n^c]$ can be constructed by hardening the calculated cover matrix $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_n]$ by

$$\mathbf{u}_k^c = \mathbf{e_m}, \ m = \arg\max_j u_{jk}, \quad j = [c], k = [n], \tag{2.34}$$

where $\mathbf{e_m}$ denotes an $m \times 1$ unity vector with 1 at its mth entry.

In the following section, the convergence of the proposed ADAL algorithm is investigated by looking at community detection in a synthetic network. Moreover, the final values of the modularity are discussed for both synthetic and real-world networks.

2.4 DISCUSSION

2.4.1 Experiments and analysis

Simulations were conducted to investigate the performance of the proposed method for community detection. In order to show the result of the convergence of proposed algorithm, first we apply the algorithm to a very simple, synthetic network with $n = 25$ and $c = 4$. Then, a brief discussion on the convergence of the proposed method is given. Finally, we applied our community detection method to several real-world data sets as described in Table 2.2. The crisp modularity values are computed and compared with FMM/GA [1], a leading modularity maximization algorithm to date, and Li's LAG [80], G-N [73], LM [81], Mincut [82], FN [83], Ncut [84] and S-A algorithms [85]. Moreover, *visual assessment of tendency* (VAT) [86] visualizations are presented to visualize the best found cover matrix.

Figure 2.1 shows the synthetic network (S) with $n = 25$ and $c = 4$. This network contains different types of node and vertex combinations, such as bridge nodes $(n_1, n_6, n_7, n_{10}, n_{11}, n_{12}, n_{16}, n_{18}, n_{19}, n_{25})$, nodes with high centrality (n_8, n_9), and low connected nodes (n_{23}, n_{24}, n_{25}). Applying the proposed method to this network, the convergence was investigated. Figure 2.2 shows the eigenvalues of $\mathbf{X}$, the output of

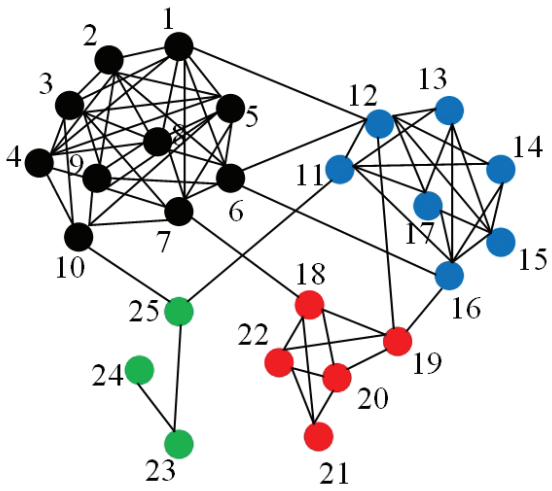


Figure 2.1: Synthetic network, $n = 25$ and $c = 4$

Algorithm 2, at the 10th and 25th iterations, respectively. As shown in Figure 2.2, most of the eigenvalues are small, as compared to the dominant eigenvalues; nevertheless, the rate of dominance increases with the number of iterations. This indicates that, by iterating Algorithm 2 together with Algorithm 3, $\mathbf{X}$ converges to a low-rank matrix, while not violating the objective function and its constraints.

Table 2.2
Real-world networks used in experiments

Name	Abbrev.	$\|E\|$	$n = \|V\|$	Network Description
Karate	K	78	34	Zachary's karate club[87]
Dolphin	D	159	62	Dolphin social network[88]
LesMis	L	254	77	Co-appearances of char. in Les Miserables[89]
Jazz	J	2742	198	Jazz musicians network[90]

Table 2.3 contains the modularity values of the cover matrix found by several leading modularity maximization algorithms and our proposed algorithm. The proposed CPP method has equal modularity value with the FMM/GA [1] for the Karate, Dolphin, Jazz, and LesMis. data sets. Moreover, the obtained modularity values are equal

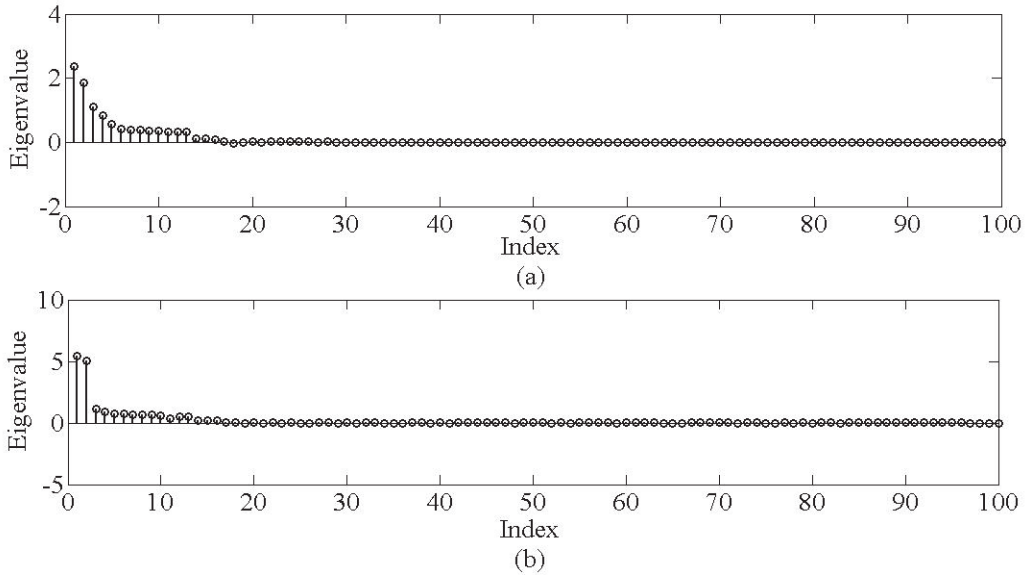


Figure 2.2: Eigenvalues of **X**, at (a) 10th iteration, and (b) 25th iteration of CPP algorithm on synthetic network.

or slightly better than the other methods: CMDR [91], G-N [73], S-A [85], Mincut [82] and GA [92]. Since the proposed method initializes the modularity values to zero, it converges to the same results given specific parameters such as ζ and μ. However, applying different values for ζ and μ may lead to different modularity values. Therefore, to evaluate the precision of the proposed CPP method, the average and the standard deviation of the modularity values are presented in Table 2.5, using fixed values of $\zeta = 0.5$ and random μ values in the interval, $0.5 \leq \mu \leq 0.99$. In [74], the author discussed applying an update rule for μ. Although this may lead to faster convergence, we found that such a method was not necessary. The applied numerical CPP parameters which lead to the best modularity values in Table 2.3 are $\zeta = 0.1$ and $\mu = 0.86$ for all data sets.

Table 2.3
Best modularity values of communities found by several algorithms on real-world data sets

Net.	c	CMDR	Mincut	LAG	FN	G-N	S-A	Ncut	FMM/GA	GA	CPP
K	4	0.4174	0.23	0.42	0.253	0.40	0.42	0.34	**0.44**	0.42	**0.44**
D	5	**0.52**	0.37	**0.52**	0.372	**0.52**	**0.52**	0.37	**0.52**	**0.52**	**0.52**
L	6	–	–	**0.56**	–	0.54	**0.56**	–	**0.56**	–	**0.56**
J	4	–	–	**0.44**	–	0.40	**0.44**	–	**0.44**	–	**0.44**

Table 2.4
Running time (second) for several algorithms for two real-world data sets

Methods	K	D
Mincut [82]	0.008	0.01
FMM/GA [1]	1295	3783
FN [83]	0.031	0.078
Ncut [84]	0.021	0.027
CPP	100.93	265.821

Table 2.4 contains the running time of corresponding modularity values proposed in CPP method. Simulations are implemented on MATLAB and executed on a Corei7, @3.4GHZ CPU and 8GBytes of RAM desktop computer.

Figure 2.3 shows the Visual Assessment of Tendency (VAT) [86] representation of the communities found by the FMM/GA algorithm [1] and the proposed CPP method for the synthetic network shown in Figure 2.1. As depicted by the numbers in each block, the detected communities found by each algorithm are the same; they are also equivalent to the expected communities shown in Figure 2.1. Figure 2.4 shows the VAT visualizations of the communities found by FMM/GA [1] and proposed CPP method for the Karate networks. It can be observed for the Karate network, the algorithms finds the same community structure (also known to be the overall max modularity community structure for Karate).

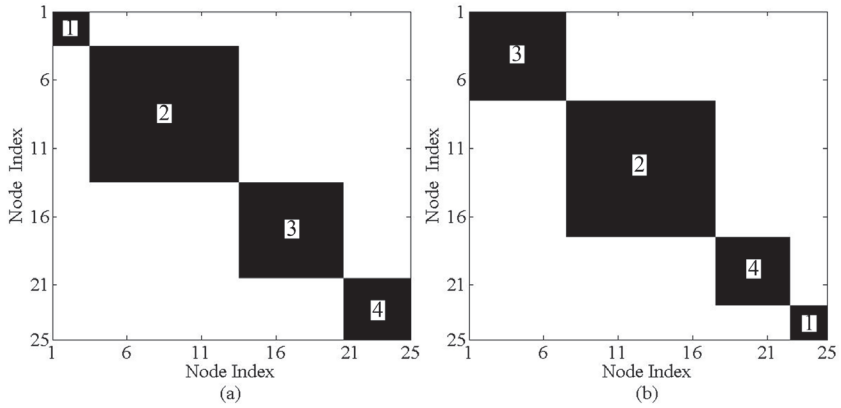


Figure 2.3: VAT visualization of communities found in synthetic network in Figure 2.1. Block numbers indicate matching clusters across algorithms. (a) FMM/GA [1], (b) CPP method.

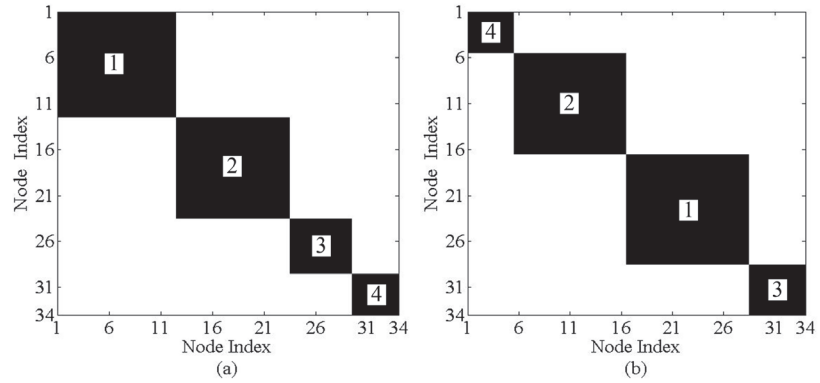


Figure 2.4: VAT visualization of communities found in Karate network. Block numbers indicate matching clusters across algorithms. (a) FMM/GA [1], (b) CPP method.

Table 2.5
Average (AVG) and standard deviation (STD) of modularity values over 200 runs

Network	**S**	**K**	**D**	**J**	**L**
CPP (AVG)	0.46	0.39	0.50	0.35	0.45
CPP (STD)	0	0.07	0.03	0.05	0.04

2.4.2 Scalability and limitations

Maximizing modularity based on a linear objective mathematical model and analysis of network communities in large data sets remain open challenges for the proposed model. The main computational hurdle is the eigen-decomposition for satisfying the rank-1 constraint. Therefore we have a need for algorithmic acceleration in this operation. The eigen-decomposition of an $n \times n$ square matrix has time complexity $O(n^3)$ [93]. Thus, in this work we will not claim that we have completely solved this NP-hard optimization problem for large-scale networks. Instead, we stress that the main motivation of this work is to show a successful reformulation of the modularity maximization problem from the viewpoint of linear programming with a completely positive constraint, which is significantly different from the existing models in terms of the objective function. It also should be noted that we are particularly interested in discovering more efficient algorithms for dealing with large scale networks, which is of big interest in the field. For large networks, new approaches should be employed to satisfy the rank-1 constraint which are faster than our proposed rank minimization algorithm. Therefore the main implementation issue is that the efficiency of CPP depends on satisfying the rank-1 constraint. This constraint is also the only bottleneck in our algorithm—the remaining steps of the algorithm are very efficient. Another point that should also be stated is that the CPP algorithm is proposed by emphasizing more the computational accuracy than the computational scalability, as compared

Table 2.6
Best modularity values of fuzzy communities found by Nepusz algorithm on real-world data sets

Network	**K**	**D**	**L**	**J**
Modularity value	0.13	0.26	0.08	0.25

to parallel algorithms for community detection in massive networks [94]. The experiments we performed using both synthetic and real-world networks have shown that, in terms of modularity value, the accuracy of the CPP algorithm is competitive with other well-known methods. We believe there is a trade-off between maximizing modularity and computational time for community detection. Such work as [33] emphasize efficiency in discovering the community structure, but at a much poorer modularity value compared with existing methods. Table 2.6 shows the modularity values found by the algorithm proposed by Nepusz et al. [33]. This algorithm is very fast, but performs poorly in terms of modularity. Our aim in the future is to find compromise between the optimal algorithm we have proposed and the efficient approach in [33].

Summary of Completely Positive Programming In this study, the modularity maximization problem for crisp community detection was addressed. We transformed the constrained quadratic problem to a linear programming problem, where the solution was constrained to the intersection of the completely positive and rank-1 cones. Hence, we took the original non-convex problem and produced a (mostly) convex optimization solution, where all but the rank-1 constraint was convex. To solve the new

objective, we used the Alternating Direction Augmented Lagrangian method. Experiments on synthetic and real-world network data showed that our method is superior to state-of-the-art modularity maximization methods at finding max-modularity crisp communities in networks. Proof of copyright permission for the necessary publications used in this dissertation are provided in Appendix C.

Chapter 3

Supper Fast Community Detection via Hybrid Label Propagation

The material in this chapter is submitted for publication in PHYSICAL REVIEW E covering statistical, nonlinear, biological, and soft matter physics on September 17, 2018.

3.1 Introduction

Most of the recent works on community detection are based on maximization of the modularity objective function proposed by Newman [23, 24, 25, 26, 31]. Other works have proposed other objective functions and solved them using extrema optimization [27], genetic algorithms [1, 28], simulated annealing [29, 35], *expectation-maximization* (EM) [30] and convex optimization [32]. More discussion on state-of-the-art approaches in community detection is available in [95].

Although some of the proposed techniques can produce acceptable performance in terms of modularity, they cannot be applied to large networks due to their high computational complexity. Raghavan et al. proposed the linear-time method based on *label propagation* (LP) [2], which can be considered as a new category for community detection that focuses on large or even massively-sized networks. The idea is to update the label of each vertex by selecting the most frequent label among its neighbors. However, it has been shown that this approach converges to local minima [96] and performance suffers due to the random vertex selection. In [35], the Newman's modularity function was optimized using simulated annealing. Although this technique performs well for small networks, it is too complex for large networks. Several versions of LP have been proposed in order to improve its performance or attain more computational efficiency [38, 97, 98, 99, 100, 101]. Some works [97, 98]

exploited the synchronized label propagation. Moreover, some other works replaced the quantitative cost function by a qualitative cost function, such as vertex importance [99, 100] or a modularity-based cost function [38, 101, 102]. However, most of these techniques are not stable, especially for large complex networks [103, 104]. Recently, some other works [105, 106] proposed the *multi-label propagation algorithm* (MLPA), where each vertex in a network can take multiple labels. However, these techniques are not efficient due to high computational complexity and required memory space, especially for large and massive networks. Blondel et al. [3] proposed a fast greedy hierarchical clustering algorithm based on LP called Louvain, showing higher performance by substituting detected communities with super-nodes at each iteration by modularity optimization technique. This algorithm has proven to produce good community structures with complexity of $O(N \log(N))$. This work inspired others to propose modified versions of Blondel's technique [40, 107]. Substituting the detected communities at each iteration with super-nodes dramatically decreases the computationally complexity by reducing the size of network. However, the Louvain technique is still complex at its first iteration, where every single node in the network is optimized subject to all available labels at its neighbors.

Network analysis is a very well researched topic in graph theory. Recently, community discovery for complex networks has drawn numerous attention. Community is a prominent structure in networks which refers to group of nodes that happens to have more connections (edge) among themselves relative to edges that connect them to the

rest of the network. Community detection and graph clustering are categorized as NP-complete problems with no globally optimal solution [108]. Applications of community detection include social network analysis, online commodity recommendation system, user clustering, biology, communication networks analysis, engineering, economics, etc. Over the decades numerous community detection algorithms have been proposed. Modularity-based clustering is one of the most prominent approaches in community detection [17]. Modularity based techniques optimize a quality objective function such as *modularity* introduced by Newman and Girvan [34] with respect to networks community. According to Newman's criteria, communities leading to higher modularity value have denser connections between the nodes within them compared to nodes of other communities [23, 26, 109].

Non-modularity based techniques are a category of community detection. Meyerhenke et al. [110] proposed high quality graph partitioning by using a parallel evolutionary algorithm to the coarsest graph. Recently, Qiao et al. [111] introduced approximate optimization to achieve parallel community detection for complex networks. Other works, such as [28, 29, 32, 35], have used non-modularity clustering objective functions to find the best graph clusters. Recently, Berahmand et al. [112] proposed a new fast local clustering approach called ECES which is based on the detection and expansion of core nodes through extended local similarity of nodes.

Although modularity based techniques provide competitive results, some of them,

such as [1, 105, 106], are not feasible approach for huge networks due to high computational complexity. Among existing methods, label propagation (LP) [2] introduced a high performance computationally efficient community discovery algorithm for large scale networks. In order to improve the efficiency of LP approach, several modified versions of LP have been introduced [38, 98, 99, 100]. The authors in [2] considered the quantity of labels, or the weighted quantity based on the adjacency matrix [38, 100] and label influence [113]. However, the best results (in terms of modularity) are presented where each label is evaluated by its impact on the modularity improvement [3, 40, 107]. Blondel et al. [3] proposes an extremely-fast high-performance unsupervised modularity-based clustering technique by applying LP in two iterative phases, known as the Louvain algorithm. Although the Louvain approach and all other modified versions [40, 43, 44, 107] propose robust performance, they still suffer from high computational complexity. In [3, 40, 107] a modified version was exploited for calculation of modularity gain achieved by label transition instead of its direct calculation, but the proposed formula is still complex for very large scale networks where millions of candidate labels should be evaluated.

Here, a novel LP-based technique is proposed that gives stable and superior solutions in terms of modularity and computational complexity. Similar to the well-known LP-based techniques, in our algorithm the optimum label for a vertex is selected from labels of its neighbors by maximizing the modularity variation associated with each label transition. Although most LP-based techniques leverage efficient approaches for

calculation of modularity variation at each transition [3, 40, 43, 44], these methods are still computationally complex in large networks, where there are hundreds or thousands of candidate labels per node to be evaluated for each node. Evaluation of modularity variation for all available labels dramatically increases computational complexity of the overall community detection procedure. Here, a novel method for label propagation based on optimizing the Newman's modularity variation associated with each transition is proposed. Instead of calculating the actual value of modularity gain corresponding to each label transition, a novel objective function corresponding to all candidate labels are developed. The proposed objective function is simplified into two terms, called the *static* and *dynamic* components. The static component represents the computationally complex term and is calculated via a static label list. However, the dynamic component represents the computationally more-efficient term and is calculated via a dynamic label list. The proposed *Hybrid Label Propagation* (HLP) technique leverages the pre-calculated values of the static component once per each iteration, while the dynamic component is calculated per each candidate label. This dramatically reduces the overall computational complexity associated with the proposed objective function.

The performance of the proposed HLP technique is evaluated over real world data sets including millions of nodes and compared with state-of-the-art techniques such as the original LP algorithm [2], Danon [35] and Louvain [3]. Moreover, HLP is compared with the non-modularity based technique, Infomap [114]. The obtained

results show that the proposed HLP technique produces competitive performance in terms of modularity value in a more computationally efficient manner, as compared to existing techniques over small to large size real-world data sets.

The rest of the paper is organized as follows. Section 3.2 introduces the mathematical model of the community detection problem and discusses the LP method and traditional modularity variation function. The proposed HLP technique is presented in Section 3.3. Section 3.4 presents experiments, analysis, and discussions, Section 3.4.2 describes HLP efficiency analysis.Table 3.1 describes select symbols and notations used throughout out this article.

3.2 Community Detection

3.2.1 Community detection and modularity

Graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$ represents a network, where V is a set of N vertices (nodes), E is a set of edges, and $\mathbf{W}$ is an $N \times N$ edge weight (or adjacency) matrix. Here, the ith row and jth column of $\mathbf{W}$, w_{ij}, denotes the weight associated with the edge connecting vertices i and j, for $i, j = [N]$. The process of community detection aims to find a $c \times N$ partition (cover) matrix $\mathbf{U}$, where the element in the ith row and the jth column in $\mathbf{U}$, u_{ki}, for $k = [c]$ and $i = [N]$, represents the membership of the

Table 3.1
Notation and symbols

Symbol	Description
$\mathbf{G}$	graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$ of network including N nodes
$\mathbf{Q}$	Modularity maximization function introduced by Newman and Girvan [34]
N	total number of nodes
$\mathbf{U}$	partition matrix of network (graph) $\mathbf{G}$
$\mathbf{u}_i$	ith column of the partition matrix $\mathbf{U}$
u_{pq}	element at pth row and qth column of $\mathbf{U}$
$\mathbf{U}_{li}$	partition matrix of G denoting l as label of the ith node
$\tilde{\mathbf{U}}_n$	partition matrix $\mathbf{U}$ with all zeros at the nth column
$\mathbf{W}$	adjacency matrix of network G
m_n	degree of the nth vertex
$\mathbf{m}$	degree vector $\mathbf{m} = [m_1, m_2, ..., m_n]^T$
$\mathbf{B}$	modularity matrix, $\mathbf{B} = \mathbf{W} - \mathbf{m}^T\mathbf{m}/\sum \mathbf{m}$
$\mathbf{b}_n$	nth column of modularity matrix $\mathbf{B}$
b_{li}	element at lth row and ith column of $\mathbf{B}$
$\mathcal{N}_i$	the set of neighbor's labels of the ith node
$\mathcal{L}'_i$	the set of labels at vicinity of the ith node
$\mathcal{L}_i$	$\mathcal{L}'_i \cup \ell_i$
c_i	number of communities at the lth iteration $\mathbf{U}$
$\mathbf{l}_s$	the static labels list
$\mathbf{l}_d$	the dynamic labels list
K	total number of iterations
$[N]$	the set of integers from 1 to N
$\setminus \{i\}$	remove ith element from the integer set
k	positive integer number

ith vertex in the kth community. In LP-based techniques, *non-overlapping* partitions prevail. Non-overlapping partitions are denoted as $\mathbf{U}$, such that

$$u_{ki} \in \{0, 1\}, \quad \sum_{k=1}^{c} u_{ki} = 1. \tag{3.1}$$

Much work has been done on non-overlapping community detection, i.e., searching

for the best $\mathbf{U}$ satisfying (3.1). In this work, HLP is proposed which exploits a novel and computationally efficient scheme to associate the optimal community (label) to each vertex corresponding to the best achievable modularity gain. The modularity value is a metric to evaluate the correctness of an associated community represented by a partition $\mathbf{U}$ [27]. Modularity was originally introduced by Newman and Girvan [34] as a way to evaluate non-overlapping communities in networks, and is defined as

$$\mathcal{Q} = \frac{1}{\|\mathbf{W}\|} \sum_{i=1,j=1}^{N} \left(w_{ij} - \frac{m_i m_j}{\|\mathbf{W}\|} \right) \delta(i,j), \tag{3.2}$$

where $m_i = \sum_{j=1}^{N} w_{ij}$, $i = [N]$, $\|\mathbf{W}\| = \sum_{i=1}^{N} m_i$, and $\delta(i,j) = 1$ if vertex i and vertex j are in the same community, else $\delta(i,j) = 0$. Liu [29] proposed a modified modularity and combined it with a simulated annealing approach for overlapping community detection. Later, Havens et al. [36] introduced a more generalized modularity, given at (3.3), that works for evaluating not only non-overlapping partitions, but also overlapping partitions.

$$\mathcal{Q}_g = \frac{\operatorname{tr}\left(\mathbf{U}\mathbf{B}\mathbf{U}^T\right)}{\|\mathbf{W}\|}, \tag{3.3}$$

where $\mathbf{B} = \left[\mathbf{W} - \frac{\mathbf{m}^T\mathbf{m}}{\|\mathbf{W}\|}\right]$ is the so called the modularity matrix for $\mathbf{m} = (m_1, m_2, \ldots, m_N)^T$ for $m_i = \sum_{j=1}^{N} w_{ij}$, $i = [N]$. Considering the generalized modularity at (3.3), a novel equation for evaluation of modularity variation is proposed. The proposed equation is simple to implement and computationally efficient

compared to the existing state-of-the-art [3, 40, 107].

3.2.2 Label propagation

The LP algorithm starts with an initialization phase where each vertex in the graph is allocated a unique label representing its community. Hence, for graph $G = (V, E)$, where V is a set of N vertices and E is a set of edges, there would be N unique labels at initialization. Then, the main body of the LP algorithm starts with an iterative process where at each iteration all labels of the graph vertices are updated. The main idea of LP is to select the *best* label for each vertex among the set of labels of its neighbors. This iterative process continues until no further improvement in the modularity variation.

Consider the label of the ith vertex at the kth iteration of LP algorithm, represented by $\ell_k(i)$; its update is performed by

$$\ell_k(i) = f(\ell_k(\mathcal{N}_i)), \tag{3.4}$$

where f is the function which selects the best label among the input labels and $\mathcal{N}_i$ is the set of labels corresponding to the neighbors of the ith vertex. Different approaches are proposed to select the best label among the available labels. Some works [2, 97, 103] considered the quantity of labels, or the weighted quantity based

on the adjacency matrix [38, 100, 101] and label influence [113]. However, the best results (in terms of modularity) are produced when each label is evaluated by its impact on the modularity improvement [3, 40, 107]. The computational complexity of the network modularity calculation for each label transition is the main drawback of this approach, especially for large and massive networks. Nevertheless, some works applied a new approach to calculating modularity gain, such as [3]

$$\begin{aligned}\Delta Q\left(i, p \rightarrow q\right) = & \left[\frac{\Sigma_{in}+k_{i,in}}{\|\mathbf{W}\|}-\left(\frac{\Sigma_{tot}+k_i}{\|\mathbf{W}\|}\right)^2\right] \\ & +\left[\frac{\Sigma_{in}}{\|\mathbf{W}\|}-\left(\frac{\Sigma_{tot}}{\|\mathbf{W}\|}\right)^2-\left(\frac{k_i}{\|\mathbf{W}\|}\right)^2\right],\end{aligned} \tag{3.5}$$

where Σ_{in} is sum of the weights of between nodes labeled q, Σ_{tot} is the sum of the weights of the nodes labeled q, k_i is the sum of the weights of node i, $k_{i,in}$ is the sum of the weights of the links from node i to nodes labeled q and $\|\mathbf{W}\|$ is defined at (3.2). Recently, a modified version of modularity variation was proposed by [40],

$$\Delta Q\left(i, p \rightarrow q\right) = \frac{\sigma\left(i, q \backslash\{i\}\right)-\sigma\left(i, p \backslash\{i\}\right)}{\|\mathbf{W}\|}+\frac{\left(\Sigma\left(p \backslash\{i\}\right)-\Sigma\left(q \backslash\{i\}\right)\right) v_i}{2\|\mathbf{W}\|^2}, \tag{3.6}$$

where $\Delta Q\left(i, p \rightarrow q\right)$ is the acquired modularity variations by re-labeling the ith node from p to q, and $\sigma\left(i, p\right)=\sum_{n,j:\ell(j)=p} w_{n,j}$, $v_i=\sum_{n,j:j\in\mathcal{N}_i} w_{n,j}+2w_{n,n}$, and $\Sigma\left(p\right)=\sum_{\forall j\ \ell(j)=p} v_j$.

Considering (3.5) or (3.6), the objective function to select the best label for the ith

vertex at the kth iteration of the LP algorithm is

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \left\{ \Delta Q \left(i, \ell_i^{(k)} \to \ell_j \right) \right\}, \forall j \in \mathcal{N}_i, \tag{3.7}$$

where $\Delta Q\left(i, p \to q\right)$ is defined in (3.5) or (3.6) and ℓ_j and $\mathcal{N}_i$ represent the label of the jth vertex and the set of neighbors of the ith node, respectively. Another important parameter in the LP algorithm is the updating sequence which is usually determined by random permutation of the N network vertices at each iteration. Here, the term *iteration* denotes a cycle where all nodes in the network have been evaluated for re-labeling using (3.7). The community or label transition process can be applied synchronously [97] or asynchronously [2, 3, 38, 100, 101], where in synchronous label transition all nodes are updated at the end of each iteration, and in asynchronous label transition nodes are updated immediately. Here, a hybrid scheme to track label transitions is considered. At the beginning of each iteration, a static label list is updated based on label transitions of the previous iteration. The static label list is exploited for computation of static components of modularity variation corresponding to each label transition. Meanwhile, each label transition is tracked and stored in the dynamic label list which is utilized for calculation of dynamic components of modularity variation for each label transition.

3.3 Proposed Hybrid Label Propagation

In this section the HLP approach is explained in detail. First, the objective function corresponding to modularity variation attained by label transition is proposed in Section 3.3.1. Then, the HLP technique is discussed in Section 3.3.2.

3.3.1 Modularity variation objective function

Considering (3.3), the modularity variation attained by changing the community (label) of the ith vertex from the current label ℓ_i to the new label ℓ_j is

$$\Delta Q\left(i, \ell_i \to \ell_j\right) = \frac{\operatorname{tr}\left(\mathbf{U}_{\ell_i \to \ell_j} \mathbf{B} \mathbf{U}_{\ell_\mathbf{i} \to \ell_\mathbf{j}}{}^T - \mathbf{U}_{\ell_i} \mathbf{B} \mathbf{U}_{\ell_i}^T\right)}{\|\mathbf{W}\|}, \tag{3.8}$$

where $\mathbf{B}$ is defined at (3.3), and $\mathbf{U}_{\ell_i \to \ell_j}$ represents the partition matrix $\mathbf{U}$ defined at (3.2), when the community label of the ith node is replaced by the label of the jth node. Also, $\mathbf{U}_{\ell_i}$ implies that the community partition does not change for the ith node. Substituting (3.8) into (3.7) leads to

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \left\{ \operatorname{tr}(\mathbf{U}_{\ell_i \to \ell_j} \mathbf{B} \mathbf{U}_{\ell_\mathbf{i} \to \ell_\mathbf{j}}{}^T - \mathbf{U}_{\ell_i} \mathbf{B} \mathbf{U}_{\ell_i}^T) \right\},\ \ell_j \in \mathcal{L}_i', \tag{3.9}$$

where $\mathcal{L}'_i$ is the set of candidate labels in the vicinity of the ith node such that $\mathcal{L}'_i = \{\ell_j,\ \forall j \in \mathcal{N}_i\}$. Defining $\mathbf{U}_{\ell_i \to \ell_j} = \tilde{\mathbf{U}}_i + \check{\mathbf{U}}_{\ell_i \to \ell_j}$, where

$$\tilde{\mathbf{U}}_i = [\mathbf{u}_1, ..., \mathbf{u}_{i-1}, \mathbf{0}_{c\times 1}, \mathbf{u}_{i+1}, ..., \mathbf{u}_N] \tag{3.10a}$$

$$\check{\mathbf{U}}_{\ell_i \to \ell_j} = [\mathbf{0}_{c\times 1}, ..., \mathbf{0}_{c\times 1}, \mathbf{e}(j), \mathbf{0}_{c\times 1}, ..., \mathbf{0}_{c\times 1}]\,, \tag{3.10b}$$

and $\mathbf{e}(j)$ denotes a $\mathbf{c} \times \mathbf{1}$ vector where $e_i(j) = 1$ for $i = j$ and $e_i(j) = 0$ for $i \neq j$. Thus

$$\begin{aligned}
\arg\max_{\ell_j} \Big\{ \mathrm{tr}(\mathbf{U}_{\ell_i \to \ell_j} \mathbf{B} {\mathbf{U}_{\ell_\mathbf{i} \to \ell_\mathbf{j}}}^T - \mathbf{U}_{\ell_i} \mathbf{B} \mathbf{U}_{\ell_i}^T) \Big\} = \\
\arg\max_{\ell_j} \Big\{ \mathrm{tr} \Big(\tilde{\mathbf{U}}_i \mathbf{B} \tilde{\mathbf{U}}_i^T + \tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T + \\
\check{\mathbf{U}}_{\ell_i \to \ell_j} \mathbf{B} \tilde{\mathbf{U}}_i^T + \check{\mathbf{U}}_{\ell_i \to \ell_j} \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T - \tilde{\mathbf{U}}_i \mathbf{B} \tilde{\mathbf{U}}_i^T - \\
\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i}^T - \check{\mathbf{U}}_{\ell_i} \mathbf{B} \tilde{\mathbf{U}}_i^T - \check{\mathbf{U}}_{\ell_i} \mathbf{B} \check{\mathbf{U}}_{\ell_i}^T \Big) \Big\},\ \ell_j \in \mathcal{L}'_i.
\end{aligned} \tag{3.11}$$

Considering $\mathrm{tr}(\mathbf{A}) = \mathrm{tr}(\mathbf{A}^T)$ and $\mathrm{tr}\left(\check{\mathbf{U}}_{\ell_i \to \ell_j} \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T\right) = \mathrm{tr}\left(\check{\mathbf{U}}_{\ell_i} \mathbf{B} \check{\mathbf{U}}_{\ell_i}^T\right) = b_{ii}$, which does not change regarding to ℓ_j, (3.11) can be simplified to

$$\arg\max_{\ell_j} \left\{ 2\mathrm{tr} \left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T - \tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i}^T \right) \right\}, \ell_j \in \mathcal{L}'_i. \tag{3.12}$$

The second component at (3.12) is constant and does not change regarding to ℓ_j, leading to

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \left\{ \text{tr}\left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T \right) \right\}, \ell_j \in \mathcal{L}_i'. \tag{3.13}$$

It should be noted that (3.13) maximizes label transitions from ℓ_i into ℓ_j for $\ell_j \in \mathcal{L}_i'$ which can lead to lower overall modularity. Therefore, in order to prevent transitions with negative impact (i.e., overall modularity gain), the proposed objective function at (3.13) must be evaluated for the current label ℓ_i as well,

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \left\{ \text{tr}\left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T \right) \right\}, \ell_j \in \mathcal{L}_i, \tag{3.14}$$

where $\mathcal{L}_i$ represents the union of the current label of the ith node and $\mathcal{L}_i'$ (the set of candidate labels at the vicinity of the ith node), i.e., $\mathcal{L}_i = \{\ell_i, \ell_j, \ \forall j \in \mathcal{N}_i\}$.

using $\text{tr}\left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T \right) = \sum_{n \setminus \{i\}} u_{\ell_j n} b_{ni}$ and the definition of the modularity matrix $\mathbf{B}$, substituting $b_{ni} = w_{ni} - \frac{m_n n_i}{\|\mathbf{W}\|}$ into (3.14) leading to

$$\text{tr}\left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T \right) = \sum_{n \setminus \{i\}} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \sum_{n \setminus \{i\}} u_{\ell_j n} m_n. \tag{3.15}$$

Applying $w_{ni} = 0$ for $n \notin \mathcal{N}_i$, the modularity variation objective function can be reduced to

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \left\{ \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \sum_{n \setminus \{i\}} u_{\ell_j n} m_n \right\}, \ \ell_j \in \mathcal{L}_i, \tag{3.16}$$

In (3.16), the first sum $S_{1,\ell_j,i} = \sum_{n\in\mathcal{N}_i} u_{\ell_j n} w_{jn}$ aggregates the edge weights corresponding to those nodes in the vicinity of the ith node labeled as ℓ_j. That is a simple process which requires search over $|\mathcal{N}_i|$ elements and $|\{k \in \mathcal{N}_i, \text{ and } \ell_k = \ell_j\}| - 1$ summations per each candidate label for each vertex. However, the second sum $S_{2,\ell_j,i} = \sum_{n\setminus\{i\}} u_{\ell_j n} m_n$ is more complex as it requires search over all nodes to find those labeled as ℓ_j, then aggregates their corresponding m. That entails search over N elements and $|\{k \in [N], \text{ and } \ell_k = \ell_j\}| - 1$ summations per each candidate label for each vertex. Therefore, the computational complexity of the second sum is much more than the first sum, especially for large N. Here, the pre-calculated values of $S_{2,\ell_j,i}$ for all available labels are exploited. However, $S_{2,\ell_j,i}$ depends on either ℓ_j and i which makes it very computational complex and also expensive to save for large networks. Hence, the element excluding notation ($\setminus\{i\}$) is removed from $S_{2,\ell_j,i}$. This does not affect $S_{2,\ell_j,i}$ for $\ell_j \neq \ell_i$ as $u_{\ell_j n} = 0$. However, for $\ell = \ell_i$, the impact of one additional i which is added by removing the element excluding notation ($\setminus\{i\}$) from $S_{2,\ell_j,i}$, must be subtracted from the modified $S_{2,\ell_j,i}$. Therefore, the final form of our modularity variation objective function is

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \begin{cases} S_{1,\ell_j,i} - \frac{m_i}{\|\mathbf{W}\|} S_{2,\ell_j}, \forall \ell_j, \ j \in \mathcal{N}_i, \\ S_{1,\ell_j,i} - \frac{m_i}{\|\mathbf{W}\|} \left(S_{2,\ell_j} - m_i\right), \ \ell_j = \ell_i, \end{cases} \tag{3.17}$$

where

$$S_{1,\ell_j,i} = \sum_{n\in\mathcal{N}_i} u_{\ell_j n} w_{jn}, \quad S_{2,\ell_j} = \sum_{n} u_{\ell_j n} m_n. \tag{3.18}$$

3.3.2 Hybrid label propagation

The HLP algorithm leverages the foundations of standard LP techniques [2], as described in Section 3.2.2, with three main differences. First, the efficient modularity variation objective function proposed at (3.17) is exploited. Second, the hybrid form of label transitions, using static and dynamic label lists is utilized. Finally, a centralized update order is instantiated where each node and its neighbors are updated sequentially at each iteration.

Algorithm 4 details the proposed HLP technique. The algorithm starts with initialization of the dynamic label list $\mathbf{l}_d$ by random distribution of labels (line 2 in Algorithm 4). Once the dynamic label list is initialized, the update list $\mathbf{q}$ is constructed (line 3 in Algorithm 4), where $q_i \in \{1, 2, ..., N\}$. The first element of $\mathbf{q}$, denoted as q_1, is picked randomly, while its corresponding neighbors are concatenated right after q_1 in the updating order vector. Then, the second node is picked (randomly from remaining nodes) and added to $\mathbf{q}$ and its neighbors that have not already been included in $\mathbf{q}$ are concatenated. The procedure is followed until all nodes are added into the updating order vector. This process is called the *centralized update order.*

The iterative procedure (lines 4-20 in Algorithm 4) starts by moving the dynamic label list $\mathbf{l}_d$ into the static label list $\mathbf{l}_s$. Then, the static label list $\mathbf{l}_s$ is used to calculate

$S_{2,\ell}$ as described at line 7 in Algorithm 4. Once $S_{2,\ell}$ is calculated for all available labels in $\mathbf{l}_s$, the label propagation procedure starts by evaluation of the modularity variation objective function corresponding to available labels of the node's neighbors ($\forall \ell_j, j \in \mathcal{N}_i$) and node's current label (ℓ_i), using (3.17) as shown in Algorithm 4 lines 10-16.

Once the modularity variations corresponding to all candidate labels are evaluated, the best label is selected and the dynamic label list $\mathbf{l}_d$ is updated (line 17 in Algorithm 4). After all vertices have been updated, the number of remaining labels is tallied as c_{l+1} for the next iteration (line 19 in Algorithm 4).

The main novelties of our proposed HLP are the use our modularity variation objective function at (3.17) instead of computing the actual value of modularity variation, and the use of the static version of the label list to calculate its second component S_{2,ℓ_j}, while the dynamic labels list is updating at each iteration and is utilized in calculation of the dynamic component $S_{1,\ell_j,i}$. In the next section, community detection results on a benchmark network and real world data sets are discussed to reveal the efficiency and feasibility of the proposed HLP technique over existing community detection approaches, such as standard LP [2], Danon [35], Infomap [114], and state-of-the-art techniques such as Louvain [3], over small, medium, and large networks.

Algorithm 4: Hybrid Label Propagation(HLP)

Require: adjacency matrix $\mathbf{W}$; initial no. of communities c_1

return $\mathbf{l}_d$
Uniformly distribute c_1 labels in $\mathbf{l}_d$,
Construct update order list $\mathbf{q}$
for $k = 1, ..., K$ **do**
  $\mathbf{l}_d \rightarrow \mathbf{l}_s$
  for $\ell = 1, 2, ..., c_l$ **do**
    $S_{2,\ell} = \sum_n u_{\ell n} m_n$, using $\mathbf{l}_s$.
  end for
  for $i \in q_1, q_2, ..., q_N$ **do**
    for $j \in \mathcal{N}_i$ **do**
      if ℓ_j is equal to ℓ_i **then**
        $\Delta Q_{\ell_j} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \left(S_{2,\ell_j} - m_i \right)$
      else
        $\Delta Q_{\ell_j} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} S_{2,\ell_j}$
      end if
    end for
    Update $\mathbf{l}_d(i) = \underset{\ell_j}{argmax} \{\Delta Q_\ell\}$,
  end for
  $c_{l+1} = |\mathbf{l}_d|$
end for

3.4 Experimental Results and Discussion

Experimental results are conducted to investigate the overall performance of the proposed HLP technique in terms of clustering performance (Newman's modularity and the number of communities), stability (standard deviation of modularity value), and scalability (processing time) for a variety of real world data sets. Moreover, the evaluation procedure is applied to leading state-of-the-art community detection techniques

Table 3.2
Parameters of LFR networks

Network	N	C_{min}	C_{max}	k_{avg}	k_{max}	γ	β	μ
LFR1	1000	10	50	25	100	2	1	$[0.1 - 0.9]$
LFR2	10000	20	100	50	200	2	1	$[0.1 - 0.9]$

to provide a comparison with our proposed HLP algorithm for the purpose of comparison. Here, the undirected and unweighted benchmark network called Lancichinetti-Fortunato-Radicchi (LFR) [115, 116] is selected to evaluate performance of the proposed technique for a network with known community structure. Table 3.2 describes the important parameter for LFR synthetic networks.

LFR benchmarks are capable to generate networks which contain common feature with the real-world data sets by assigning various values to the parameters represented by the Table 3.2. Here N denotes the number of nodes; C_{min} is the number of the nodes within the smallest community; C_{max} is the number of the nodes within the largest community; k_{avg} is the average degree of the nodes; k_{max} is the maximum degree of the nodes in the network; γ is the exponent for the degree sequence and β is the exponent for the community size distribution; also μ is a mixing parameter coefficient in the LFR network denotes the average rate of edges that connect nodes from different communities. That means by increasing value of μ, the strength of the community structure decreases and makes the community detection more difficult.

To this end, *normalized mutual information* (NMI) [117], which lies in the ranges $[0, 1]$, has been employed to quantify the ability of the algorithm to discover the

known community structure. An NMI = 0 represents two independent partitions, were NMI = 1 refers to identical partitions. For the real-world data sets, modularity Q is used to measure the quality of the detected communities. Duch et al [27] showed that higher modularity value represents more accurate detected partition. Meanwhile, NMI is evaluated for two ground-truth networks, Dolphin and Football.

3.4.1 Efficiency analysis

Table 3.3 contains the characteristics of the real-world networks used in the experiments and the parameters used for HLP. Different network sizes were selected to explore the performance and the scalability of the proposed technique compared to the state-of-the-art methods such as Louvain [3], Danon [35], classic LP [2] and Infomap [114]. Here, the Infomap approach is selected as a non-modularity based state-of-the-art technique with which to compare. Modularity and NMI are selected as for evaluation of the detected communities. The experiments are run on the same machine to develop a fair comparison of running time. The experiments are coded and executed in MATLAB (Danon, LP, Louvain and HLP) and C (Infomap), using a laptop incorporating an i7-6560U processor @2.20GHz with 16GB of memory.

Tables 3.4 and 3.5 shows the average modularity values, processing time, and the number of detected communities for the real-world data sets. Here, it is observed that

Table 3.3
Network characteristics and parameters used in HLP

Network	Nodes	Edges	c_1	Iteration K	Ground Truth
Dolphin [88]	62	159	50	5	Yes
Football [118]	115	613	50	5	Yes
Jazz [90]	198	2742	50	5	No
Metabolic [27]	453	4,596	50	5	No
Email [119]	1,133	5451	50	5	No
Ego-Facebook [120]	4,039	88,234	100	5	No
Email-Enron [121]	36,692	183,831	2000	5	No
Com-dblp [121]	317,080	1,049,866	20000	5	Yes
Com-youtube [121]	1,134,890	2,987,624	20000	5	Yes

the proposed HLP technique produces competitive results as compared to Louvain [3], Danon [35], and classic LP [2] techniques in terms of average modularity value, but in a much shorter processing time. Although Infomap [114] seems faster, it should be considered that the proposed processing time in Table 3.5 corresponds to C implementation which is usually faster than MATLAB, which was used for simulation of the other methods. Moreover, it is observed that the Infomap technique leads to smaller modularity value for large networks such as Com-dblp and Youtube data sets. Authors in [122] illustrated that Infomap suffers from the field-of-view limit for large communities and fails to detect large communities.

Comparing the number of detected communities, it is observed that the HLP method converges to a smaller number of communities in comparison to the classic LP and Infomap. This could be seen as a desirable result, as it may help further analysis and interpretation of the features or characteristics of the communities [123, 124]. These result would be expected as the propagation of labels in original LP happens more

Table 3.4
Experiment results over real-world data sets–part 1, average Newman's modularity Q_m, processing time in sec t_s and the number of detected communities C

Algorithm	Louvain			Danon			LP	
Network	Q_m	t_s	C	Q_m	t_s	C	Q_m	t_s
Dolphin	0.519	0.102	5	0.5136	0.0038	4	0.4994	0.0137
Football	0.604	0.105	9	0.5714	0.011	6.7	0.5769	0.0274
Jazz	0.443	0.152	3	0.4393	0.0317	3	0.4316	0.067
Metabolic	0.424	0.131	9	0.413	0.193	9.3	0.3845	0.0985
Email	0.540	0.379	11	0.5408	4.6	10.8	0.4927	0.2941
Ego-Facebook	0.8323	3.86	15.60	0.8124	382	13	0.8154	2.67
Email-Enron	0.5845	54.58	1185.3	***	$\gg 10e6$	***	0.5572	50.24
Com-dblp	0.8099	1186.8	143.4	***	$\gg 10e6$	***	0.684	2664.5
Com-youtub	0.6987	21082	3262.4	***	$\gg 10e6$	***	0.6235	42351

often due to the impact of each label transition into the calculation of modularity variations. Meanwhile, in HLP, the label transitions are not applied into the static label list which is used for calculation of modularity variations. However, the Louvain approach usually converges to fewer communities as it combines detected nodes within communities to construct a super-node. That prevents label transition of each node at former iterations and leads to fewer communities.

Tables 3.6 and 3.7 contains the standard deviation of Newman's modularity and the number of detected communities evaluated over 100 runs (10 runs for Com-YouTube network). Compared to the modularity score itself, the modularity standard deviation is very low for all the community detection algorithms. The proposed HLP does produce a slightly higher standard deviation than the others, though the difference overall is negligible. As it is shown next, the HLP essentially trades a slightly degraded clustering performance and standard deviation for a significant boost in scalability

Table 3.5
Experiment results over real-world data sets–part 2, average Newman's modularity Q_m, processing time in sec t_s and the number of detected communities C

Algorithm	LP	Infomap			HLP		
Network	C	Q_m	t_s	C	Q_m	t_s	C
Dolphin	8.85	0.5259	0.0064	5	0.5106	0.0109	7.25
Football	11.5	0.6032	0.008	11	0.5721	0.0213	9.12
Jazz	6.3	0.4421	0.0209	5	0.4427	0.0506	5.59
Metabolic	37.5	0.4061	0.0283	27	0.4032	0.0607	25.47
Email	46.3	0.5367	0.073	48	0.5113	0.1472	28.17
Ego-Facebook	59.8	0.7051	0.525	6	0.8051	0.9362	35.58
Email-Enron	1831.7	0.5293	3.08	1077	0.5535	7.4865	1143
Com-dblp	19637	0.671	31.5	24318	0.6953	243.02	5564.6
Com-youtub	19838	0.557	148.6	23466	0.6113	5902.3	13308

Table 3.6
Experiment results over real-world data sets standard deviation of Newman's modularity (σ_Q) and the number of detected communities (σ_C)-Part-1

Algorithm	Louvain [3]		Danon [35]		Original LP [2]	
Network	σ_Q	σ_C	σ_Q	σ_C	σ_Q	σ_C
Dolphin	0.0091	0.5975	0	0	0.0092	0.8567
Football	0.0090	0.8591	0.0061	0.4785	0.0175	1.3304
Jazz	0.0125	0.5385	0.0020	0	0.0090	0.9241
Metabolic	0.0057	1.0724	0.0020	0.7616	0.0118	2.8334
Email	0.0068	0.9055	0.0039	0.5196	0.0138	1.7972
Ego-Facebook	0.0061	1.0909	0.0025	0.6557	0.0039	3.7868
Email-Enron	0.0091	4.6690	***	***	0.0101	5.4836
Com-dblp	0.0046	3.4496	***	***	0.0047	7.4766
Com-youtub	0.0038	9.32	***	***	0.0093	6.324

and efficiency, enabling performance for very large networks.

Tables 3.8 and 3.9 shows the NMI values for the ground-truth real world data sets, Dolphin and Football. These results demonstrate that HLP performs on par with the other community detection approaches. If we were to pick, the Louvian method is the

Table 3.7
Experiment results over real-world data sets standard deviation of Newman's modularity (σ_Q) and the number of detected communities(σ_C)-Part-2

Algorithm	Infomap [114]		HLP	
Network	σ_Q	σ_C	σ_Q	σ_C
Dolphin	0.0346	0.7759	0.0130	1.0392
Football	0.0021	0.6356	0.0205	1.4213
Jazz	0.0787	0.8075	0.0058	0.8118
Metabolic	0.0245	2.7386	0.0075	2.8100
Email	0.0073	2.8071	0.0158	3.5763
Ego-Facebook	0.0019	2.3087	0.0079	3.3690
Email-Enron	0.0048	5.4222	0.0175	5.8762
Com-dblp	0.0056	7.7910	0.0191	9.6469
Com-youtub	0.0038	11.316	0.0379	13.49

Table 3.8
The NMI of the real-world networks with ground truth communities-Part1.

Network	No. Nodes	No. Edges	No. Groups	Louvain	Danon
Dolphin	62	159	2	0.5498	0.5742
Football	115	613	12	0.9200	0.8084

Table 3.9
The NMI of the real-world networks with ground truth communities-Part-2.

Network	No. Nodes	No. Edges	No. Groups	Original LP	Infomap	HLP
Dolphin	62	159	2	0.4425	0.4880	0.4917
Football	115	613	12	0.9293	0.9522	0.9129

overall winner in this test. But the message here is that HLP is a good community detection algorithm and when the efficiency of HLP is considered, it clearly shines above the rest.

Figure 3.1 illustrates the average modularity values versus algorithm iterations for various real-world data sets. It is observed that HLP converges fast and produces a

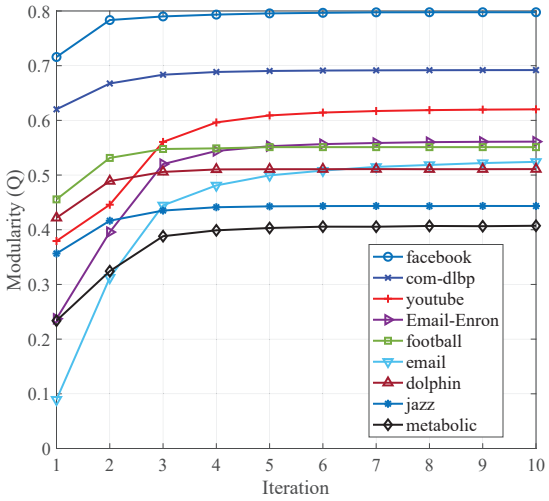


Figure 3.1: Proposed HLP modularity convergence for small to huge real data sets.

steady behavior after the 5th iteration for most of the available data sets.

Figure 3.2 depicts the impact of the initial number of communities on (a) modularity value and (b) running time for Facebook and Email-Enron networks. As shown, the modularity performance is not sensitive to the number of initial communities c_1. This can be observed in Fig. 3.2(a) where values in $20 \leq c_1 \leq 200$ for Email data set and $50 \leq c_1 \leq 200$ for Facebook network approximately lead to the same results. As expected, increasing the number of initial communities increases the computational complexity due to more available labels over which to search for each node in early iterations. For instance, increasing $c1$ for Facebook network from 10 to 100 leads to a 50% increase in overall processing time.

Figure 3.3 shows the average NMI and modularity values for the LFR network [115, 116], with 1,000 nodes and variable mixing parameters μ. An average degree of 25 and

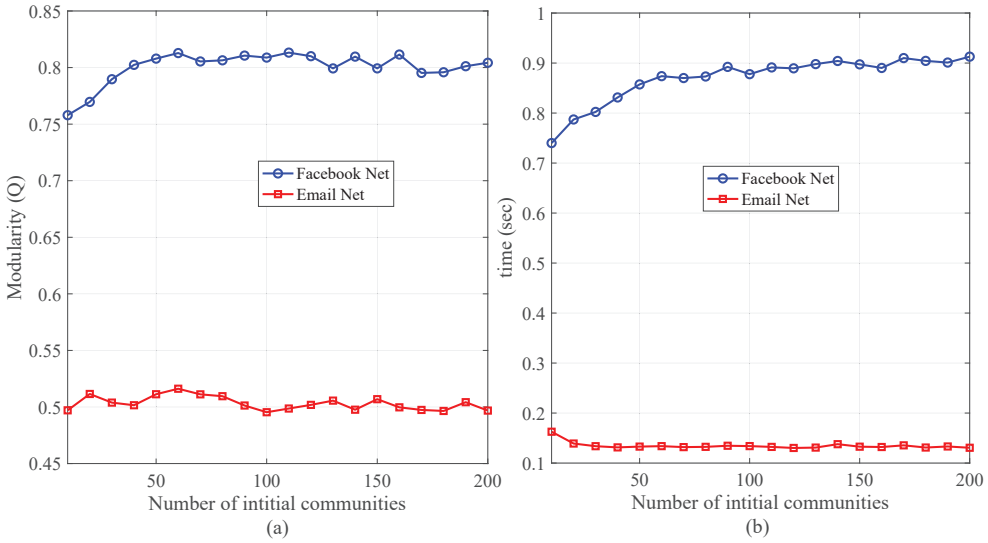


Figure 3.2: Impact of initial number of communities (c_1) on algorithm performance over 100 run(a) Average modularity value, (b) Algorithm processing time.

maximum degree of 100 were selected. As shown in Figure 3.3(a), the HLP approach leads to NMI> 0.9 for $\mu \leq 0.6$. Although the NMI value drops for $\mu > 0.6$, HLP outperforms the Infomap [114], Louvain [3], and Danon [35] techniques for $\mu > 0.6$. Comparing the modularity values shown in Figure 3.3(b), it is observed that the original LP, Louvain, Infomap, and HLP perform nearly the same for $\mu < 0.6$, while the Infomap approach fails to detect available large communities. It is also observed that by increasing mixing parameter μ and the size of the network graphs, the overall performance decreases as expected. Again, these results show that HLP is a good community detection algorithm.

Figure 3.4 shows the average NMI and modularities for the LFR network [115, 116] with 10,000 nodes and variable mixing parameters μ. An average degree of 50 and

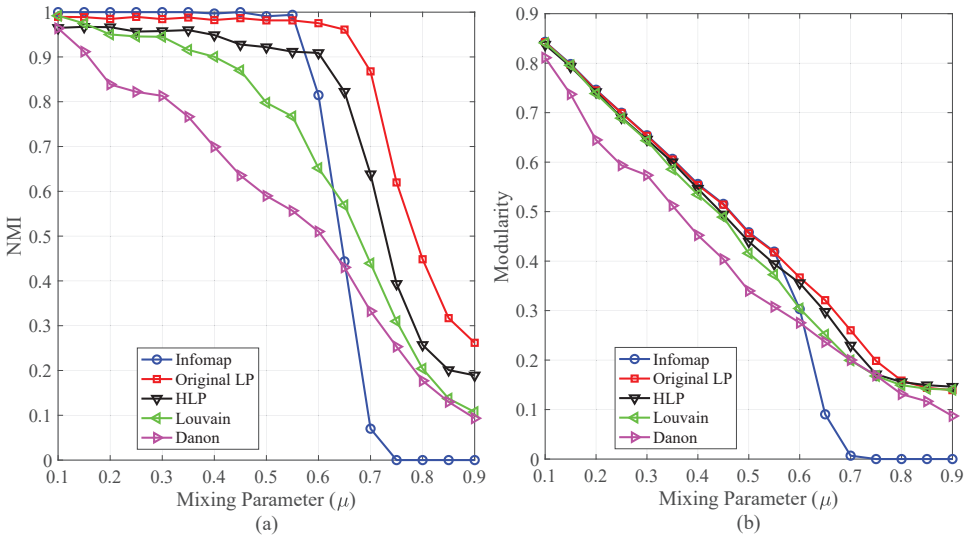


Figure 3.3: Average NMI and modularity evaluated using LFR network with 1000 nodes(LFR1).

maximum degree of 200 were selected. As shown in Figure 3.3(a), the HLP approach leads to NMI$>$ 0.9 for $\mu \leq 0.65$. Although the NMI value drops for $\mu > 0.65$, HLP outperforms the Infomap, Louvain and Danon techniques for $\mu > 0.6$. Comparing the modularity values plotted in Figure 3.4(b), it is also observed that the original LP, Louvain, Infomap, and HLP perform nearly the same for $\mu < 0.7$, while the Infomap approach fails to detect available large communities for $\mu \geq 0.8$.

Summarizing the overall results, it is observed that the HLP provides feasible community detection solutions in terms of modularity and NMI in small to huge networks. However, the main contribution of HLP is its ability to quickly partition huge networks, as shown in Tables 3.4 and 3.5. In the next section, the computational complexity of the proposed HLP approach is discussed analytically.

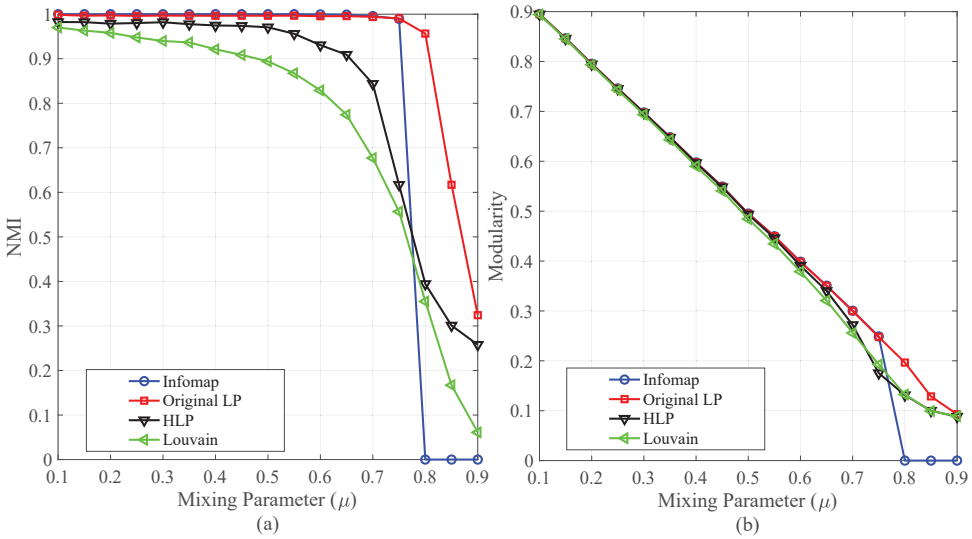


Figure 3.4: Average NMI and modularity evaluated using LFR network with 10000 nodes(LFR2).

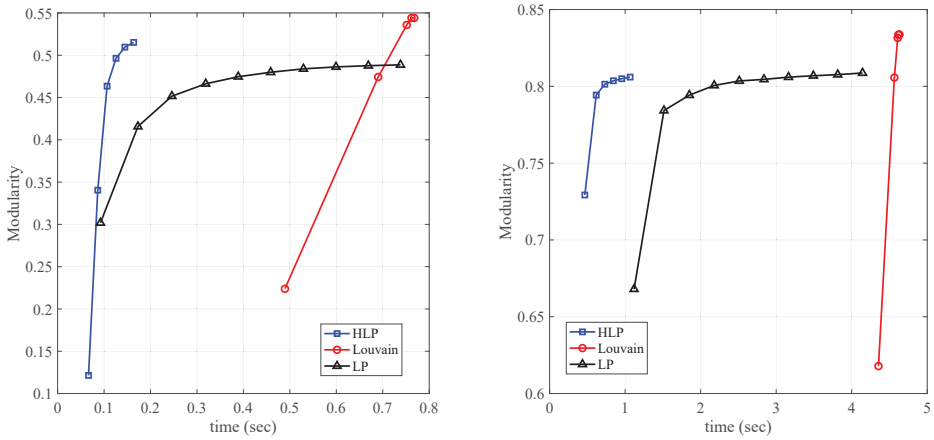
3.4.2 Computational complexity analysis

Considering (3.5) and (3.6), it is observed that the traditional methods for calculation of modularity variation require an N element search to find all nodes labeled the same as the current (p) and the candidate (q) labels. Moreover, $N_k, k \in \{p, q\}$, sums are required to calculate the total sum-weights, such as Σ_{tot} at (3.5) or $\sigma(i, k)$ at (3.6), where N_k denotes the number of nodes labeled as k within the entire network. The same procedure is required for calculation of Σ_{in} and $k_{i,in}$ at (3.5) and $\Sigma(p)$ at (3.6). These components are calculated per candidate label transition. In HLP, calculation of $S_{2,\ell}$ requires an N-element search and an N_ℓ-size sum which is executed only once

per each label, using the static label list. Moreover, calculation of the dynamic component, $S_{1,\ell,i}$, requires an $\mathcal{N}_i$-element search and $n_{i,\ell}$ sums, where $\mathcal{N}_i$ and $n_{i,\ell}$ denote the number of neighbors of the ith node and those labeled as ℓ, respectively. Therefore, it is concluded that the overall number of operations required for calculation of modularity variation for all nodes is $O(N^2)$ for the proposed approaches at (3.5) and (3.6) versus $O(N)$ for HLP.

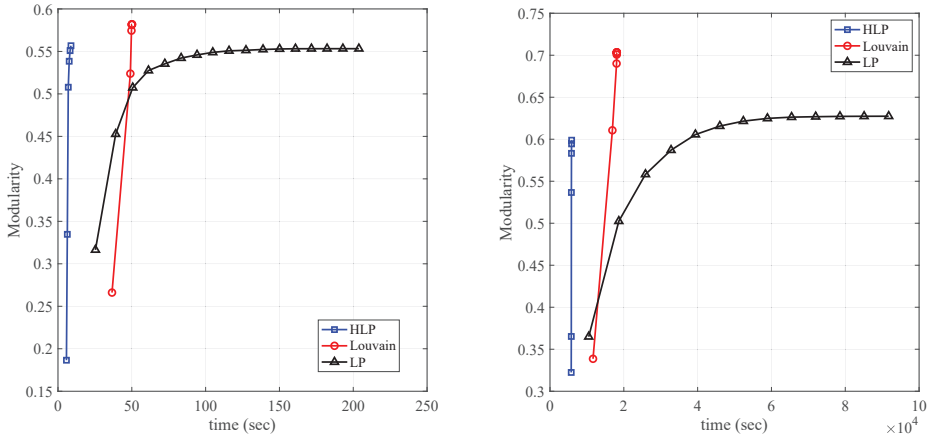
Figures 3.5 and 3.6 depict the convergence of HLP, Louvain, and original LP over multiple iterations in terms of average modularity for the Email, Facebook, Email-Enron, and YouTube data sets. As shown in Fig. 3.6, it is observed that HLP converges to the final solution prior to the first iteration of Louvain and LP methods for large networks. This is because the main computational complexity of the LP-based techniques, such as Louvain and original LP, correspond to the calculation of modularity gain variations which are dramatically decreased in HLP. Therefore, as a quantitative comparison, it can be observed that HLP always outperforms Danon, LP, and Louvain in term of processing time, and produces competitive modularity values.

Summary of Hybrid Label Propagation In this study, community detection via a novel hybrid label propagation approach was proposed. We developed a label propagation method that exploited static and dynamic labels to reduce the computation at each iteration. The proposed technique selects the optimum label by maximizing a



(a) Using non-overlapping memberships (b) Using overlapping memberships

Figure 3.5: HLP versus LP and Louvian : average modularity convergence versus time.(a) Email data set, (b) Ego-Facebook data Set.



.pdf

(a) Using non-overlapping memberships (b) Using overlapping memberships

Figure 3.6: HLP versus LP and Louvian : average modularity convergence versus time.(a) Email-Enron data set, (b) YouTube data set.

novel modularity variation objective function, which then optimizes the overall modularity gain for a given label propagation. In the proposed objective function, the

dynamic labels are used for calculating low complexity components of the objective function for each candidate label transition. Meanwhile, the static labels are used for off-line calculation of the computationally expensive components of the objective function for each available label.

It was observed that the performance of the proposed HLP was very competitive in terms of modularity and normalized mutual information as compared to existing techniques for small to medium sized networks, and with a lower computational complexity. For large networks, HLP converged to acceptable solutions (although, sometimes slightly sub-optimal) in a far faster time than existing methods. Therefore, it was concluded that the proposed HLP is a proper remedy with acceptable computation time to discover high-quality community structure in large-scale networks, where most recent existing methods are highly computationally expensive for that application and thus are not a feasible solution for massive networks.

Chapter 4

Linear Time Community Detection by a Novel Modularity Gain Acceleration in Label Propagation

The material in this chapter was submitted for publication to IEEE Transactions on Big Data on March 12, 2019.

4.1 Introduction

In this chapter a novel strategy for calculation of modularity gain associated with label transitions is discussed. The proposed approach is called *Modularity Gain Acceleration* (MGA), as it is inspired by analysis of the general closed form model of Newman's modularity shown at (4.2). Unlike the traditional method proposed in [3, 40, 43, 44, 107], here, a new formulation of the modularity gain objective function is developed corresponding to the available candidate labels. Using mathematical manipulations, the proposed objective function is simplified into two components: the *Local Sum-Weight* (LSW) and the *General Sum-Weight* (GSW). The LSW is the lower complexity component and is calculated once per each *candidate* label transition, for each node. The GSW is the computationally complex component and is calculated only once per each label at the initiation phase. However, the GSW needs an updating procedure to keep up with the network's community membership variations throughout the LP procedure. Therefore, an update scheme is conducted over the GSWs corresponding to the source and destination communities at each label transition. This update process requires only two additions, which leads to a huge efficiency gain compared to direct calculation of GSW per each label transition.

The efficiency of the proposed technique is evaluated analytically by examining the required mathematical operations and compared with traditional approaches in [3, 40,

43, 44, 107]. Moreover, the efficiency of the proposed technique is evaluated over real world data sets with millions of nodes, using the proposed MGA in state-of-the-art LP-based community detection techniques, such as Louvain [3] and traditional LP [2]. The obtained results show that the MGA produces the same performance in terms of modularity, as expected; however, it offers significant speed-up proportional to the size of network. Simulation results on a real world data set with millions of nodes demonstrates that our method outperform most existing modularity based clustering approaches in term of both time complexity and modularity performance.

The rest of the chapter is arranged as follows. Section 4.2 introduces the mathematical model of the community detection and discusses the LP method, including the traditional approach for so-called effective calculation of modularity gain variation. Then the proposed MGA technique is presented in Section 4.3 followed by an analytical evaluation of computational complexity associated with the proposed MGA and traditional approach. Section 4.4 presents experiments, analysis, and discussions.

Table 4.1 contains the selected symbols and notations used throughout this chapter.

Table 4.1
Notation and symbols

Symbol	Description
G	graph $G = (V, E, \mathbf{W})$ of network including N nodes
$\mathbf{U}$	partition matrix of network (graph) G
$\mathbf{u}_i$	ith column of the partition matrix $\mathbf{U}$
u_{li}	element at lth row and ith column of $\mathbf{U}$
$\mathbf{U}_{li}$	partition matrix of G indicating l as label of node i
$\tilde{\mathbf{U}}_n$	partition matrix $\mathbf{U}$ with all zeros at the nth column
$\mathbf{W}$	adjacency matrix of network G
m_n	degree of the nth vertex
$\mathbf{m}$	degree vector $\mathbf{m} = [m_1, m_2, ..., m_n]^T$
$\mathbf{B}$	modularity matrix
$\mathbf{b}_n$	nth column of modularity matrix $\mathbf{B}$
b_{li}	element at lth row and ith column of $\mathbf{B}$
$\mathcal{N}_i$	the set of neighbor's labels of the ith node
$\mathcal{L}'_i$	the set of candidate labels for node i
$\mathcal{L}_i$	$\mathcal{L}'_i \cup \ell_i$
$[N]$	the set of integers from 1 to N
$\setminus \{i\}$	remove ith element from the integer set

4.2 Community Detection

4.2.1 Community detection and modularity

Consider a network as an undirected graph $G = (V, E, \mathbf{W})$, where V is a set of N vertices (nodes), E is a set of edges, and $\mathbf{W}$ is an $N \times N$ adjacency matrix. Here, the ith row and jth column of $\mathbf{W}$, w_{ij}, denotes the weight associated with the edge connecting vertices i and j, for $i, j = [N]$ [1]. The process of community detection aims

[1]Note that the notation $[N]$ means the set of integers from 1 to N.

to find a $c \times N$ partition matrix $\mathbf{U}$, where the element in the kth row and ith column of $\mathbf{U}$, u_{ki}, for $k = [c]$ and $i = [N]$, represents the membership of the ith vertex in the kth community. Crisp partitions are $\mathbf{U}$, such that $u_{ki} \in \{0, 1\}$ and $\sum_{k=1}^{c} u_{ki} = 1$.

The modularity value is a metric to evaluate the correctness of an associated community [27] represented by a partition $\mathbf{U}$. It was by Newman and Girvan [34] as a metric to evaluate quality of non-overlapping communities in graph clustering, and is defined as

$$\mathcal{Q} = \frac{1}{\|\mathbf{W}\|} \sum_{i=1,j=1}^{N} \left(w_{ij} - \frac{m_i m_j}{\|\mathbf{W}\|} \right) \delta(i, j), \tag{4.1}$$

where $m_i = \sum_{j=1}^{N} w_{ij}$, $i = [N]$, $\|\mathbf{W}\| = \sum_{i=1}^{N} m_i$. $\delta(i, j) = 1$ if vertex i and vertex j are in the same community, else $\delta(i, j) = 0$. Liu et al. [29] introduced a new modularity objective function for overlapping community detection in networks. Havens et al. [36] is developed a more generalized modularity metric, given at (4.2), that works for evaluating either overlapping or non-overlapping partitions.

$$\mathcal{Q} = \frac{tr\left(\mathbf{U}\mathbf{B}\mathbf{U}^T\right)}{\|\mathbf{W}\|}, \tag{4.2}$$

where $\mathbf{B} = \left[\mathbf{W} - \frac{\mathbf{m}^T\mathbf{m}}{\|\mathbf{W}\|}\right]$ is the modularity matrix, $\mathbf{m} = (m_1, m_2, \ldots, m_N)^T$, and $m_i = \sum_{j=1}^{N} w_{ij}$, $i = [N]$. In this chapter the proposed form od modularity at (4.2) is utilized to develop a new approach for calculation of modularity gain achievable by a label transition in LP-based community detection.

4.2.2 Label propagation clustering

The LP algorithm starts with an initialization phase where each vertex in the graph is allocated a unique label representing its community. Hence, for graph $G = (V, E)$, there would be N unique labels at the initialization step. Then, the main body of the LP algorithm starts with an iterative process where at each iteration all labels of the graph vertices are updated. The LP approach selects the best label (among all available labels in a node's neighborhood) with respect to the best gain in term of modularity value that can be obtained for each candidate label transition. This iterative process will be continued until no further improvement in modularity gain. This demands evaluation gain corresponding to every candidate label. In very large scale networks, numerous numbers of label transitions have to be evaluated at each iteration, which leads to huge computational complexity.

Consider traditional modularity gain

$$\Delta Q\left(i, p \rightarrow q\right) = \left[\frac{\Sigma_{in} + k_{i,in}}{\|\mathbf{W}\|} - \left(\frac{\Sigma_{tot} + k_i}{\|\mathbf{W}\|}\right)^2\right] + \left[\frac{\Sigma_{in}}{\|\mathbf{W}\|} - \left(\frac{\Sigma_{tot}}{\|\mathbf{W}\|}\right)^2 - \left(\frac{k_i}{\|\mathbf{W}\|}\right)^2\right], \tag{4.3}$$

where Σ_{in} is sum of the weights between nodes labeled q, Σ_{tot} is the sum of the weights corresponding to nodes labeled q, k_i is the sum of the weights of node i, $k_{i,in}$ is the

sum of the weights of the links from node i to nodes labeled q and $\|\mathbf{W}\|$ is defined at (4.1). Recently, a modified version of modularity gain was proposed [40]

$$\Delta Q\left(i, p \to q\right) = \frac{\sigma\left(i, q \backslash \{i\}\right) - \sigma\left(i, p \backslash \{i\}\right)}{\|\mathbf{W}\|} + \frac{\left(\Sigma\left(p \backslash \{i\}\right) - \Sigma\left(q \backslash \{i\}\right)\right) v_i}{2\|\mathbf{W}\|^2}, \tag{4.4}$$

where $\Delta Q\left(i, p \to q\right)$ is the acquired modularity gain by re-labeling the ith node from p to q, and $\sigma\left(i, p\right) = \sum_{i,j:\ell(j)=p} w_{i,j}$, and $\Sigma\left(q\right) = \sum_{\forall i \in \nu_q} v_i$ for $v_i = \sum_{i,j:j\in\mathcal{N}_i} w_{i,j} + 2w_{i,i}$, where ν_q represents the set of all nodes labeled as q.

Considering (4.3) or (4.4), the objective function for selecting the best label for the ith vertex at the kth iteration of the LP algorithm is

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \left\{ \Delta Q\left(i, \ell_i^{(k)} \to \ell_j\right) \right\}, \forall j \in \mathcal{N}_i, \tag{4.5}$$

where $\Delta Q\left(i, p \to q\right)$ is defined at (4.3) or (4.4) and $\ell\left(j\right)$ and $\mathcal{N}_i$ represent the label of the jth vertex and the set of neighbors of the ith node, respectively.

In this work, an efficient approach called MGA is introduced which selects the best available label among the labels of neighbors, according to the new modularity gain of label transitions.

4.3 Modularity Gain Acceleration

In this section the proposed MGA approach is explained in detail. First, the objective function corresponding to the attained modularity gain by label transition is proposed. Then the computational complexity of the proposed objective function is studied analytically.

4.3.1 The MGA approach

Consider a label transition of the ith node from current label ℓ_i to the new label ℓ_j based on (4.2); the modularity gain ΔQ is obtained by (see Appendix B for proof)

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \Bigg\{ \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \sum_{n \setminus \{i\}} u_{\ell_j n} m_n \Bigg\}, \ell_j \in \mathcal{L}_i, \tag{4.6}$$

where the first sum, $S_{1,\ell_j,i} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{jn}$, represents the LSW, which aggregates the edge weights corresponding to those nodes of the neighbors of the ith node labeled as ℓ_j. That is a low computationally complex process which requires search over $|\mathcal{N}_i|$ elements and $(|\{k \in \mathcal{N}_i, \text{ and } \ell_k = \ell_j\}| - 1)$ summations per each candidate label for each node. However, the second sum, $S_{2,\ell_j,i} = \sum_{n \setminus \{i\}} u_{\ell_j n} m_n$, represents the GSW, is

a significant time consuming process as it requires search over all nodes to find those labeled as ℓ_j, then summation of their corresponding m which requires to search over N elements and $(|\{k \in [N], \text{ and } \ell_k = \ell_j\}| - 1)$ summations per each candidate label for each vertex. Here, the pre-calculated values of GSWs or $S_{2,\ell_j,i}$ are exploited for all available labels. However, $S_{2,\ell_j,i}$ depends on either ℓ_j and i which makes it very computationally complex and also expensive to save for large scale networks.

Here, we propose to use simple mathematical manipulations as follow to remove the node index subscript i for the proposed GSW. To this end, we need to remove the element excluding notation ($\backslash\{i\}$) from $S_{2,\ell_j,i}$ or $\sum_{n\backslash\{i\}} u_{\ell_j n} m_n$. This does not affect $S_{2,\ell_j,i}$ for $\ell_j \neq \ell_i$ as $u_{\ell_j i} = 0$. However, for $\ell_j = \ell_i$, the impact of one additional i which is added by removing the element excluding notation ($\backslash\{i\}$) from $S_{2,\ell_j,i}$, must be subtracted from the modified $S_{2,\ell_j,i}$. Thus, the modularity objective gain function can be simplified to

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \begin{cases} S_{1,\ell_j,i} - \frac{m_i}{\|\mathbf{W}\|} S_{2,\ell_j}, & \forall \ell_j, j \in \mathcal{N}_i, \\ S_{1,\ell_j,i} - \frac{m_i}{\|\mathbf{W}\|} \left(S_{2,\ell_j} - m_i\right), & \ell_j = \ell_i, \end{cases} \tag{4.7}$$

where

$$S_{1,\ell_j,i} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{jn}, \tag{4.8a}$$

$$S_{2,\ell_j} = \sum_{n} u_{\ell_j n} m_n, \tag{4.8b}$$

As mentioned, (4.8b) is only computed once at the initialization stage. However, by propagating labels throughout the LP process, the membership of vertices are subjected to change. Therefore, the pre-calculated values of GSWs proposed at (4.7) are no longer valid. This problem is addressed by an efficient updating stage, which compensates for the impact of each label transition through the procedure. Considering label transition of the ith node from the ℓ_ith to the ℓ_jth community, the following update rule must be applied to the pre-calculated GSWs corresponding to ℓ_i and ℓ_j:

$$S_{2,\ell_j} + m_i \rightarrow S_{2,\ell_j}, \tag{4.9a}$$

$$S_{2,\ell_i} - m_i \rightarrow S_{2,\ell_i}, \tag{4.9b}$$

where S_{2,ℓ_i} and S_{2,ℓ_j} represent the GSWs corresponding to the old (ℓ_i) and the new (ℓ_j) labels associated to the ith node, respectively, and m_i is defined at (4.1).

The proposed equations at (4.7) and (4.8) along with the updating equations at (4.9) present the main contribution of this chapter. The main novelty of this work is reforming the GSW such that the node subscripts are removed, which allows off-line calculation of the GSW per each label (S_{2,ℓ_j}) following by an update process instead of on-line calculation of the GSW for all available labels of all nodes ($S_{2,\ell_j,i}$). Algorithms 5 and 6 detail the process of deploying the MGA technique into the original LP [2] and Louvain [3], respectively.

Algorithm 5: MGA Label Propagation (MGA-LP)

Require: adjacency matrix $\mathbf{W}$; initial no. of communities c_1

return label list $\mathbf{l}$
initialize vertices communities $\ell = 1, 2, ..., c_l$
$S_{2,\ell} = \sum_n u_{\ell n} m_n$, using initialized communities.
while $Q_{new} > Q_{old}$ **do**
  for $i = 1, 2, ..., N$ **do**
    for $j \in \mathcal{N}_i$ **do**
      if ℓ_j is equal to ℓ_i **then**
        $\Delta Q_{\ell_j} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \left(S_{2,\ell_j} - m_i \right)$
      else
        $\Delta Q_{\ell_j} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} S_{2,\ell_j}$
      end if
    end for
    Update $\mathbf{l}(i) = \arg\max_{\ell_j} \{\Delta Q_\ell\}$,
    $S_{2,\ell_j} + m_i \rightarrow S_{2,\ell_j}$
    $S_{2,\ell_i} - m_i \rightarrow S_{2,\ell_i}$
  end for
end while

4.3.2 Computational complexity analysis

In this section the number of mathematical operations required for calculation of modularity gain variations associated with a label transition is evaluated analytically. The traditional approaches, proposed at (4.3) or (4.4), and the proposed MGA scheme are evaluated. Table 4.2 reviews the number of operations including summation, multiplication, and search required to calculate a modularity gain at (4.3) or (4.4), and the MGA approach proposed at (4.7). In (4.4), Σ_{in} and $k_{i,in}$ are the most complex components. First, N search operations must be applied to extract a set of nodes labeled as q, or ν_q, such that $\{\ell_j = q,\ \forall j \in \nu_q\}$. Then $k_j = |\mathcal{N}_j|$ searches

Algorithm 6: MGA Louvain Algorithm (MGA-Louvain)

Require: adjacency matrix $\mathbf{W}$;

return label list $\mathbf{l}$
Initialize each node as single community and set $N_c = N$
while $Q_{new} > Q_{old}$ **do**
  for $\ell = 1, 2, ..., N_c$ **do**
    $S_{2,\ell} = \sum_n u_{\ell n} m_n$, using initialized communities.
  end for
  for $i = 1, 2, ..., N_c$ **do**
    for $j \in \mathcal{N}_i$ **do**
      if ℓ_j is equal to ℓ_i **then**
        $\Delta Q_{\ell_j} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \left(S_{2,\ell_j} - m_i \right)$
      else
        $\Delta Q_{\ell_j} = \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} S_{2,\ell_j}$
      end if
    end for
    Update $\mathbf{l}(i) = \arg\max_{\ell_j} \{\Delta Q_\ell\}$,
    $S_{2,\ell_j} + m_i \to S_{2,\ell_j}$
    $S_{2,\ell_i} - m_i \to S_{2,\ell_i}$
  end for
  Set N_c with the number of available communities
  Construct supper nodes for $i = [N_c]$
  Set each supper node as single community
end while

Table 4.2
Required mathematical operations for calculation of modularity gain variations to move the ith node into the qth community.

Operation	traditional (4.3) or (4.4)	MGA (4.7)
Search	$\sum_{j \in \nu_q} \lvert\mathcal{N}_j\rvert + \lvert\mathcal{N}_i\rvert + N$	$\lvert\mathcal{N}_i\rvert$
Summation	$\sum_{j \in \nu_q} \left\lvert\mathcal{N}_j^{(q)}\right\rvert + \left\lvert\mathcal{N}_i^{(q)}\right\rvert + 4$	$\left\lvert\mathcal{N}_i^{(q)}\right\rvert + 4$
Multiplication	3	2

per each node in ν_q are needed to reveal the set of weights corresponding to the neighbors labeled q, or $w_{j,j'}^{(q)}$, such that $\{j, j' \in \nu_q,\ j' \in \mathcal{N}_j\}$. Then $\left(\left|\mathcal{N}_j^{(q)}\right| - 1\right)$ summations are needed to aggregate weights in $w_{j,j'}^{(q)}$, $\forall j \in \nu_q$. Therefore, $\sum_{j \in \nu_q} |\mathcal{N}_j|$ searches and $\sum_{j \in \nu_q} \left(\left|\mathcal{N}_j^{(q)}\right| - 1\right)$ summations are needed to calculate Σ_{in}. Moreover,

calculation of $k_{i,in}$ requires $k_i = |\mathcal{N}_i|$ search operations to reveal the set of weights corresponding to the neighbors labeled q, or $w_{i,i'}^{(q)}$, such that $\{i, i' \in \nu_q,\ i' \in \mathcal{N}_i\}$. Then, $\left(\left|\mathcal{N}_i^{(q)}\right| - 1\right)$ summations are needed to aggregate weights in $w_{i,i'}^{(q)}$. Therefore, overall $k_i = |\mathcal{N}_i|$ search operations and $\left(|\mathcal{N}_i^{(q)}| - 1\right)$ summations are needed to calculate $k_{i,in}$. Furthermore, (Σ_{tot} requires $|\nu_q| - 1$ summations to aggregate total weights of nodes labeled q. The rest of components, such as k_i and $\|\mathbf{W}\|$, are constant values and can be calculated off-line. Additionally, 6 summations and 3 multiplications are need to calculate the final value of ΔQ at (4.3).

Taking a closer look at (4.4), it is observed that the modularity gain value is derived with respect to the same components at (4.3). The $\Sigma(i, q \backslash \{i\})$ component at (4.4) represents the sum of weights among nodes in ν_q, or Σ_{in} at (4.3). Moreover, the $\sigma(i, q \backslash \{i\})$ component at (4.4) represents the weights between the ith node and nodes in ν_q, or $k_{i,in}$ at (4.3). However, the most complex part at (4.7) is the LSW or $S_{1,\ell_j,i}$ which like the $k_{i,in}$, demands $k_i = |\mathcal{N}_i|$ search operations and $\left(|\mathcal{N}_i^{(q)}| - 1\right)$ summations.

As shown in Table 4.2, traditional approaches, such as (4.3) or (4.4), have overall computational complexity that depends on the size of the network N and the size of each community or $|\nu_q|$, which usually increases with the size of network. However, for the proposed MGA approach, the overall computational complexity is on the order of the node neighborhood size, $|N_i|$, which usually depends on network topology rather

than its size. Practical evaluation of traditional approaches and MGA for real-world data sets is presented next.

4.4 Experimental Results and Discussion

Experiments are conducted to investigate the performance of the proposed MGA technique in terms of computational complexity. It should be noted that as the proposed MGA leads to the same modularity gain as the traditional method, it thus leads to the same final modularity value—and the same communities. Therefore, here the final network topology (in terms of Newman's modularity) is evaluated beside the computational complexity (in terms of processing time). The evaluation process is conducted for the classic LP [2] and the Louvain [3] techniques using the proposed MGA (Algorithms 5 and 6). In order to assess the quality of the proposed technique on real-world data sets versus state-of-the-art approaches, we also present the most recent non-LP algorithm, called ECES [112].

Table 4.3 shows the characteristics of the real-world networks used in the experiments and the parameters used for MGA over 100 runs. Here, different sizes of networks are selected to explore the performance and the scalability of the proposed technique compared to the state-of-the-art methods. The experiments are executed 100 times for each network on the same machine to develop a fair comparison. The experiments

Table 4.3
Network characteristics and parameters used in MGA

Network	Nodes	Edges	c_1 (LP)	Ground Truth
Dolphin [88]	62	159	50	Yes
Football [118]	115	613	100	Yes
Jazz [90]	198	2742	100	No
Metabolic [27]	453	4,596	100	No
Email [119]	1,133	5451	100	No
Ego-Facebook [120]	4,039	88,234	100	No
Email-Enron [121]	36,692	183,831	2000	No
Com-Dblp [121]	317,080	1,049,866	2000	Yes
Com-Youtube [121]	1,134,890	2,987,624	10000	Yes

are coded and executed in MATLAB, using a laptop with an i7-6560U processor @2.20GHz with 16GB of memory. Note that ECES [112] is implemented in ANSI C++ using a PC with an i5 CPU (2.8 GHz) and 6GB of memory.

Tables 4.4 and 4.5 show the average convergence time over 100 runs of the traditional LP and Louvain approaches versus the proposed MGA-LP and MGA-Louvain methods in Algorithm 5 and 6, respectively. Moreover, the results of the ECES [112] technique are presented to compare the classic LP and Louvain using the MGA with a state-of-the-art technique.

Figures 4.1-4.9 depict the learning curve of the traditional LP [2] and Louvain [3], versus the LP and Louvain techniques using the MGA proposed in Algorithms 5 and 6. As expected and shown in Table 4.4, the modularity values at each iteration are the same. However, it can observed that the proposed MGA-Louvain technique converges to the final solution before even the first iteration of the traditional approaches, for

Table 4.4
Average processing time over 100 run in sec t_s and average Newman's modularity Q_m for real-world data set.

Algorithm	ECES [112]		LP [2]		MGA-LP	
Network	t_s	Q_m	t_s	Q_m	t_s	Q_m
Dolphin	3	0.495	0.0184	0.4902	0.0164	0.4902
Football	5	0.549	0.0243	0.5740	0.0201	0.5740
Jazz	5	0.291	0.0847	0.437	0.0713	0.437
Metabolic	9	0.403	0.2219	0.3835	0.1428	0.3835
Email	14	0.480	0.7381	0.4886	0.4094	0.4886
Facebook	22	0.524	4.143	0.8086	1.255	0.8086
Email-Enron	150	0.517	2038.1	0.5531	22.69	0.5531
Com-dlbp	480	0.728	1,5361	0.6496	258.4	0.6496
Com-YouTube	3240	0.569	91,769	0.6273	671.4	0.6273

Table 4.5
Average processing time over 100 run in sec t_s and average Newman's modularity Q_m for real-world data set.

Algorithm	Louvain [3]		MGA-Louvain	
Network	t_s	Q_m	t_s	Q_m
Dolphin	0.102	0.519	0.0316	0.519
Football	0.105	0.604	0.0365	0.604
Jazz	0.152	0.443	0.0545	0.443
Metabolic	0.131	0.424	0.1085	0.424
Email	0.379	0.540	0.4073	0.540
Facebook	3.86	0.8323	1.232	0.8323
Email-Enron	54.58	0.5845	9.499	0.5845
Com-dlbp	1186.8	0.8099	296.2	0.8099
Com-YouTube	21082	0.6987	2,589	0.6987

$N > 1000$ (see Figures 5.6–4.9). The same results are observed for LP when $N > 4000$ (see Figures 4.6–4.9).

The most interesting result of MGA is the linearity of the computational complexity with respect to the size of the network. That makes MGA more superior to the traditional approaches when the size of the network increases. For instance, by applying

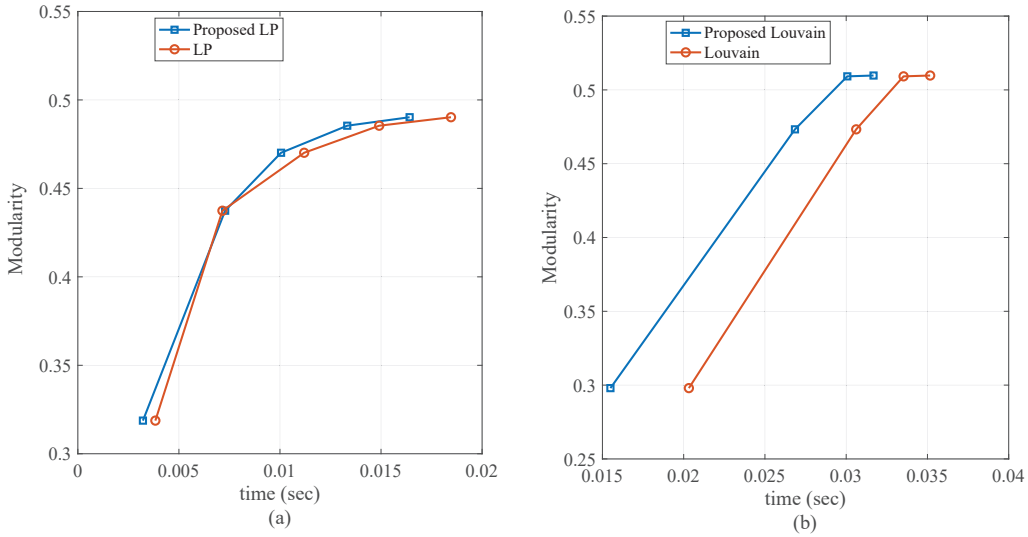


Figure 4.1: Modularity learning curve for Dolphin data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

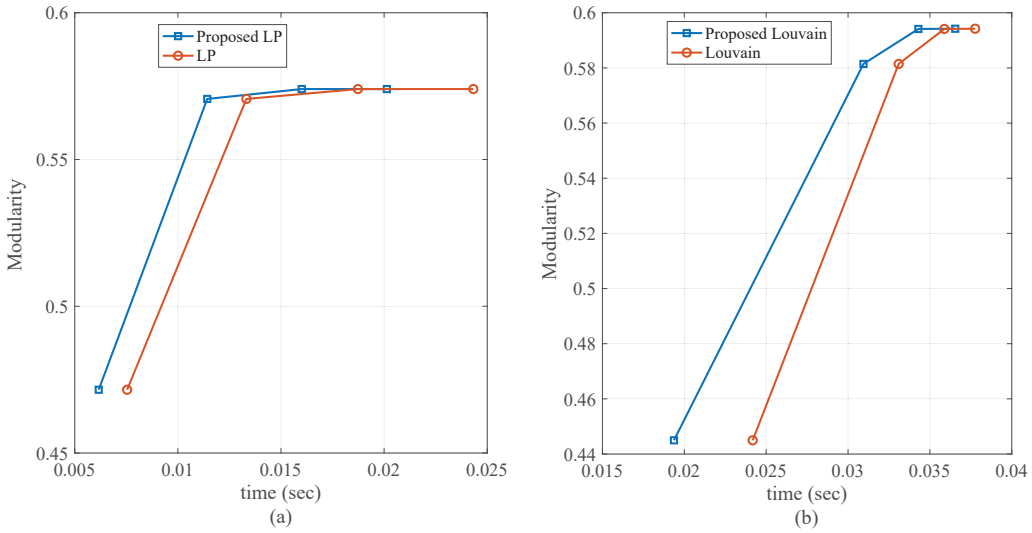


Figure 4.2: Modularity learning curve for Football data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in 5 and 6

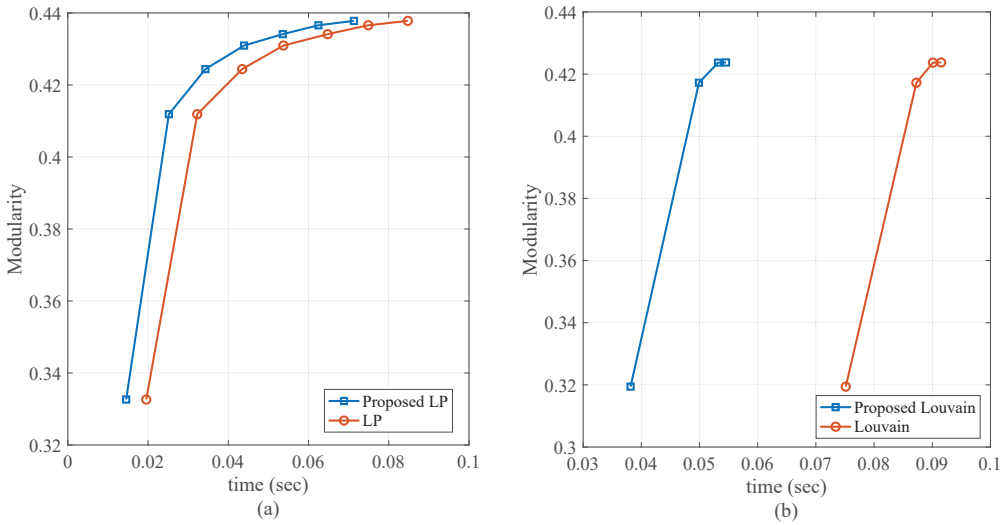


Figure 4.3: Modularity learning curve for Jazz data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

MGA, the computational complexity is reduced more than 100 times for traditional LP and about 8 times for Louvain for the YouTube data set shown in Table 4.4 and Figure 4.9. The difference between the computational time reduction between LP and Louvain in large networks is due to the size reduction that happens when constructing the super nodes in the Louvain approach. Meanwhile, the first iteration of the Louvain approach is still too complex (about 65% of overall complexity) which leads to an 85% reduction of computational complexity for a network with around 1.1 million nodes. Moreover, it is observed that the proposed MGA-Louvain approach always outperform ECES in terms of time complexity and final modularity.

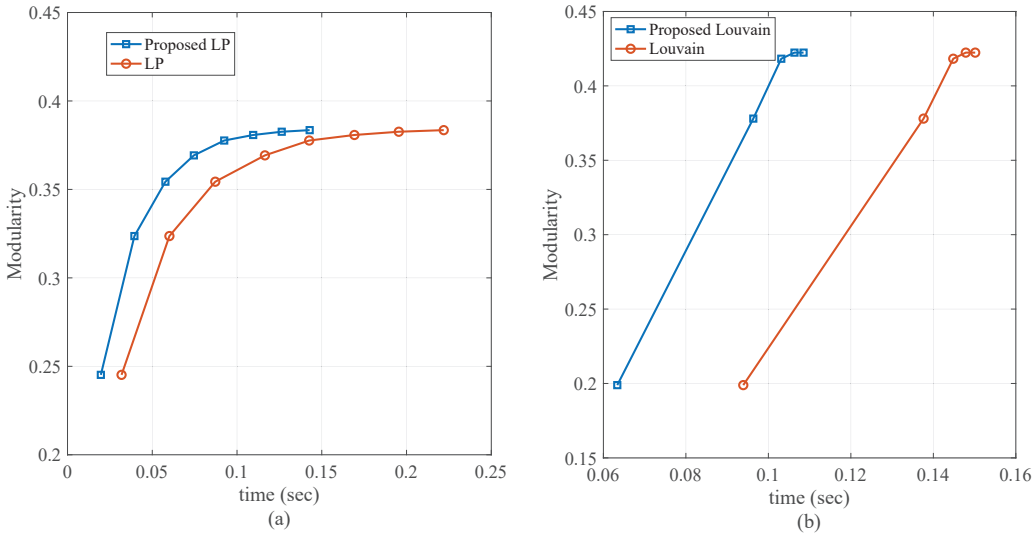


Figure 4.4: Modularity learning curve for Metabolic data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

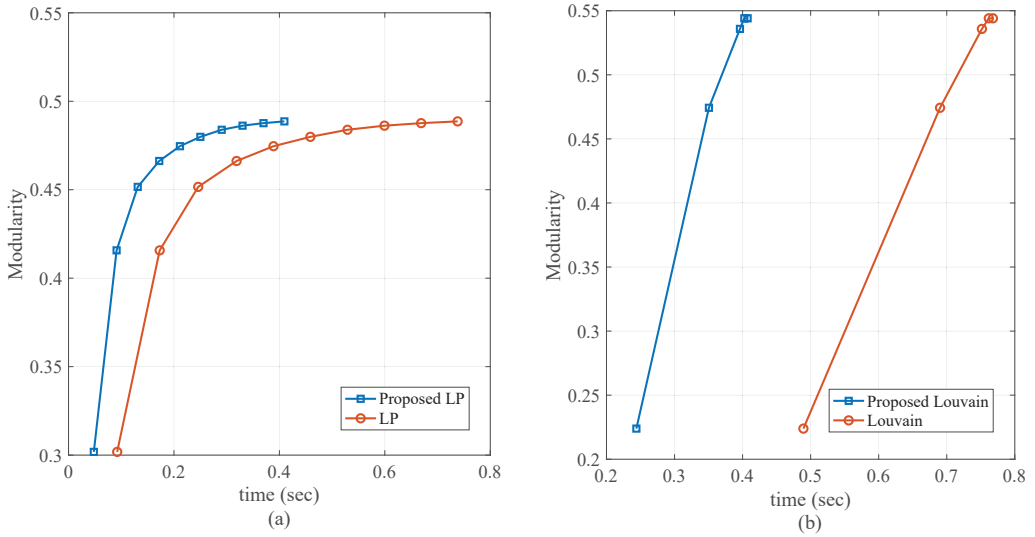


Figure 4.5: Modularity learning curve for Email data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

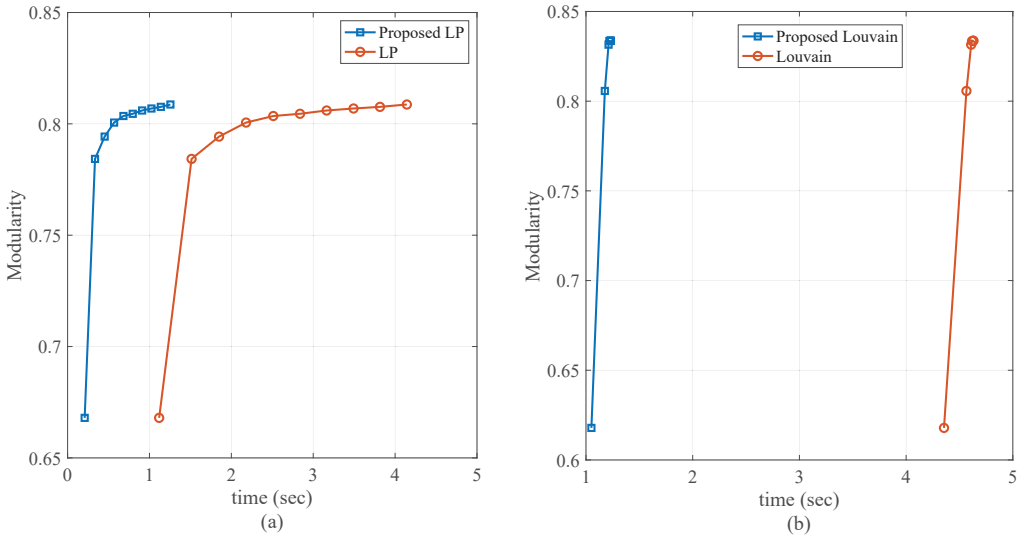


Figure 4.6: Modularity learning curve for Facebook data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

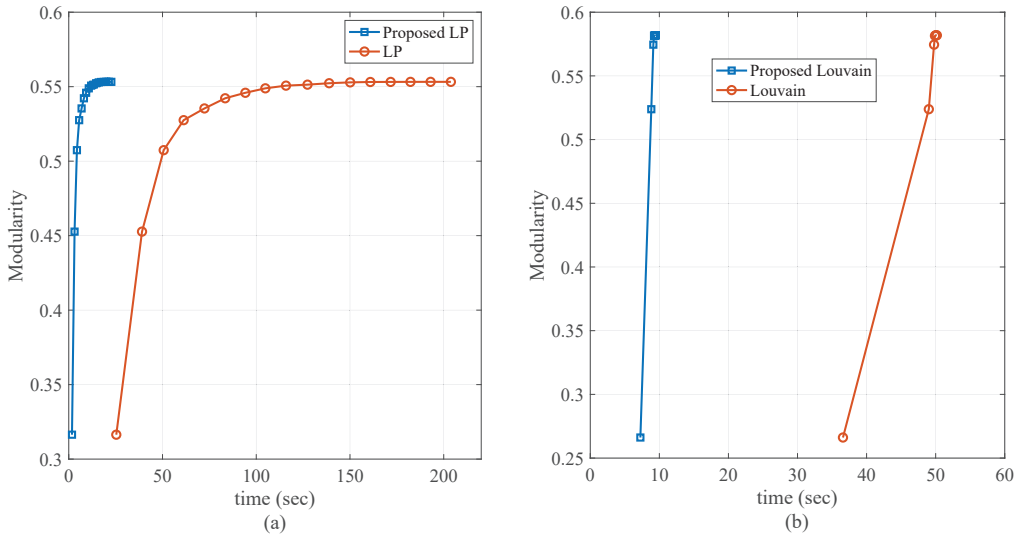


Figure 4.7: Modularity learning curve for Email-Enron data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

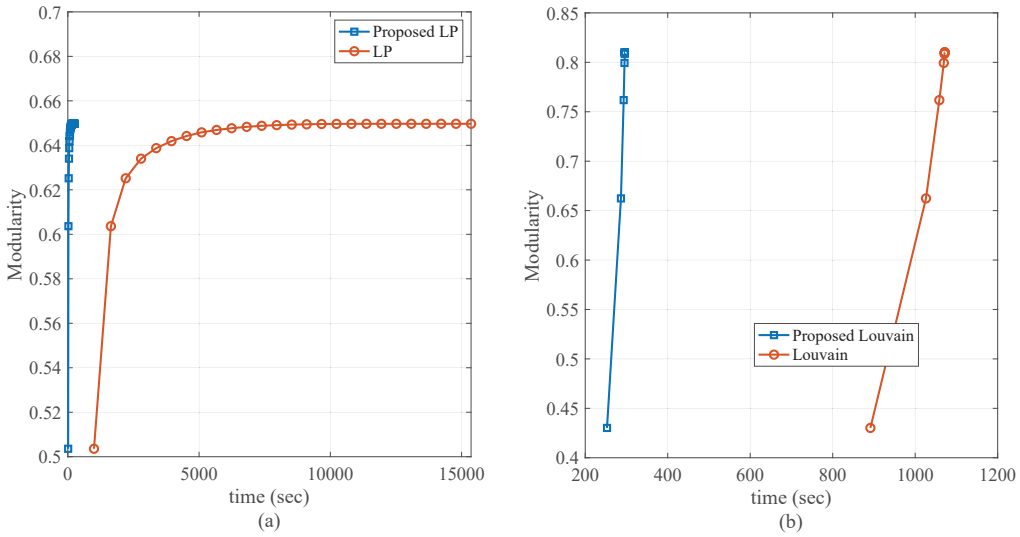


Figure 4.8: Modularity learning curve for Com-DLBP data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

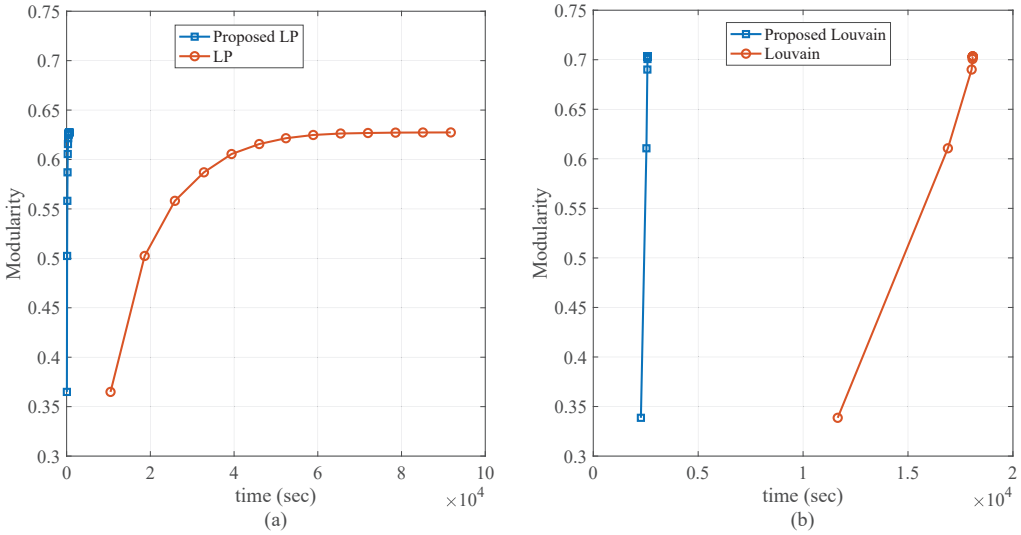


Figure 4.9: Modularity learning curve for Com-YouTube data set applying traditional LP [2] and Louvain [3] versus MGA-LP and MGA-Louvain in algorithm 5 and 6

Summary of Linear Time Modularity Gain Acceleration In this study, we introduced a novel objective function for calculation of the attained modularity gain corresponding to label transitions. The proposed technique is efficient as it offers linear computational complexity (with respect to the size of the network) associated with calculation of modularity gain variation per each vertex label transition. The computational complexity of the proposed technique was assessed analytically and compared with traditional approaches developed for the calculation of modularity gain. Then, real-world data sets, containing up to millions of nodes, were tested with two non-overlapping LP-based community detection schemes that incorporated traditional and the proposed MGA approaches for modularity gain variations. The proposed technique is applied to selected state-of-the-art LP-based community detection methods and the resulting network modularity and execution time are compared with traditional methods. By applying MGA to LP-based methods, the run-time is significantly reducedsometimes finishing before the traditional approach even finishes one iterationand the same modularity result and number of communities, i.e., community detection result, is obtained. The MGA approach leads to significant efficiency improvements by reducing time consumption up to 85% relative to the original algorithms with the exact same quality in terms of modularity value.

Chapter 5

Overlapping Community Detection in Large-Scale Complex Networks via Fast Fuzzy Modularity Maximization

The material in this chapter is in preparation for submission to IEEE Transactions on Fuzzy Systems.

5.1 Introduction

As discussed in Chapter HLP, community detection approaches are categorized as either non-overlapping or overlapping in terms of node—i.e., vertex—membership value. In non-overlapping community detection, each node belongs to only one community; meanwhile, in overlapping community detection each vertex can belong to more than one community [36]. Non-overlapping community detection has attracted a lot of attention [2, 3, 23, 24, 25, 26, 37, 38, 39, 40], and efficient approaches such as those proposed in [3, 40] were developed in terms of performance (modularity) and computational complexity. However, these techniques cannot produce true membership values corresponding to nodes with high between-ness [36]. Therefore, many works discuss overlapping community detection [4, 28, 29, 30, 31, 32, 33, 36, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134]. Although some of these techniques have acceptable performance in terms of modularity, most of them suffer from high computational complexity and only are applied to small or medium size networks [4, 31, 33, 36, 125, 126, 127, 128, 133, 134, 135].

Some works propose fast overlapping community detection [41, 42]; however, performance of their overlapping community detection is not measured in terms of modularity. Some recent works propose faster techniques for overlapping community detection via modularity maximization. In [129] the authors proposed a method based on fuzzy

label propagation which still is much more complex than non-overlapping community detection due to the higher number of available communities to be tested and high computational complexity of the fuzzy modularity change value compared to that of the non-overlapping form. In [130] the authors proposed a game theory based method which is too complex for very large networks—on the order of hundreds of minutes for networks less than 10^5 nodes. In [134], the authors proposed a *fuzzy agglomerative* (FuzAg) approach for community detection that iteratively updates membership degree of nodes. The time complexity of the FuzAg algorithm is $O(n^2)$. Note, however, that the FuzAg algorithm has impressive performance in term of modularity value for small real world data sets. Su and Havens [125] proposed several heuristics for soft modularity maximization. Similar to [134], the proposed FMM/H2 has great performance in terms of modularity value, but is only appropriate for small data sets due to its time complexity of $O(n^2)$.

In this chapter, overlapping community detection via *Fast Fuzzy Modularity Maximization* (FFMM) is proposed. First, a novel objective function for calculation of modularity gain associated with changing the membership values of each vertex (a column of the partition matrix) is introduced. Then, the FFMM technique is developed by optimizing this objective function—i.e., modularity gain—subject to node fuzzy memberships. The efficiency of the proposed objective function is enhanced by incorporating the pre-calculation of static components at each iteration. Moreover, multi-cycle FFMM is introduced which breaks networks into multiple sub-networks

and applies the FFMM to detect their communities. Then, the detected communities at each sub-network are considered as sub-networks for the next cycle.

The proposed multi-cycle FFMM technique offers remarkable performance in terms of modularity and *overlapping normalized mutual information* (ONMI) in near linear time with respect to network size and initial number of communities to be detected. Studies over real-world data sets and the *Lancichinetti-Fortunato-Radicchi* (LFR) [115, 116] benchmark network show that multi-cycle FFMM is much faster (on the order of network size) as compared to the state-of-the-art techniques, such as [125, 130, 134], with impressive modularity and remarkable ONMI values.

The rest of the chapter is organized as follows. Section 5.2 introduces the mathematical model of the community detection problem and discusses both non-overlapping and fuzzy approaches. Section 5.3 details the proposed FFMM technique. Multi-cycle FFMM is introduced in Section 5.4. Simulation results and discussions are proposed in Section 5.5. Table 5.1 contains selected notation and symbols used throughout this chapter.

Table 5.1
Notation and symbols

Symbol	description
G	graph $G = (V, E, \mathbf{W})$ of network including N nodes
$G_j^{(l)}$	graph of jth sub-network at lth cycle
$\mathbf{U}$	partition matrix of network (graph) G
$\mathbf{u}_n$	the nth column of the partition matrix $\mathbf{U}$
u_{kn}	element at kth row and nth column of $\mathbf{U}$
$\mathbf{U}_j^{(l)}$	partition matrix of jth sub-network at the lth cycle
$\mathbf{u}_{j,n}^{(l)}$	the nth column of the partition matrix $\mathbf{U}_j^{(l)}$
$u_{j,kn}^{(l)}$	element at the kth row and nth column of $\mathbf{U}_j^{(l)}$
$\tilde{\mathbf{U}}_{[n]}$	partition matrix $\mathbf{U}$ with all zeros at the nth column
$\mathbf{W}$	adjacency matrix of network G
m_n	degree of the nth vertex
$\mathbf{m}$	degree vector $\mathbf{m} = [m_1, m_2, ..., m_n]^T$
$\mathbf{B}$	modularity matrix, $\mathbf{B} = \mathbf{W} - \mathbf{m}^T\mathbf{m}/\sum \mathbf{m}$
$\mathbf{b}_n$	the nth column of modularity matrix $\mathbf{B}$
b_{kn}	element at the kth row and the nth column of $\mathbf{B}$
$\tilde{\mathbf{B}}$	modularity matrix $\mathbf{B}$ with all zero diagonal elements
$\mathbb{N}_n$	the set of neighbors of the nth node in G
C	total number of communities of $\mathbf{U}$
$C^{(l)}$	total number of communities at the lth cycle
$c^{(l)}$	resolution at the lth cycle
$c_j^{(l)}$	number of target communities of the jth sub-network at the lth cycle
$N_j^{(l)}$	number of nodes in the jth sub-network at the lth cycle
$[t]$	the set of integers from 1 to t
n_0	non-negative integer number
k	positive constant number
L	number of cycles

5.2 Community Detection

5.2.1 Modularity

Every network can be represented by a graph $G = (V, E, \mathbf{W})$, where V is a set of n vertices, E is a set of edges, and $\mathbf{W}$ is an $n \times n$ edge weight (or adjacency) matrix, where w_{ij} in $\mathbf{W}$ denotes the weight of the edge connecting vertex i and vertex j. Community detection for a network is the process of finding a $c \times n$ partition matrix $\mathbf{U}$, where each element u_{ki} in $\mathbf{U}$, $k = [c]$, $i = [n]$, is the *membership* of the ith vertex in the kth community.

The value of modularity denotes the accuracy of communities represented by a partition $\mathbf{U}$. Modularity was originally introduced by Newman and Girvan [34] as a way to evaluate non-overlapping communities in networks, which is defined as

$$\mathcal{Q} = \frac{1}{\|\mathbf{W}\|} \sum_{i=1,j=1}^{n} \left(w_{ij} - \frac{m_i m_j}{\|\mathbf{W}\|} \right) \delta(i,j), \tag{5.1}$$

where $m_i = \sum_{j=1}^{n} w_{ij}, i = [n]$, $\|\mathbf{W}\| = \sum_{i=1}^{n} m_i$, and $\delta(i,j) = 1$ if vertex i and vertex j are in the same community, else $\delta(i,j) = 0$. In this work, the generalized modularity

objective function proposed by Havens et al. [36] is used;

$$\mathcal{Q}_g = \frac{\mathrm{tr}\left(\mathbf{UBU^T}\right)}{\|\mathbf{W}\|}, \tag{5.2}$$

where $\mathbf{B} = \left[\mathbf{W} - \frac{\mathbf{m}^T\mathbf{m}}{\|\mathbf{W}\|}\right]$ and $\mathbf{m} = (m_1, m_2, \ldots, m_N)^T$ for $m_i = \sum_{j=1}^{N} w_{ij}, i = [N]$. The proposed modularity objective function at (5.2) works for evaluating not only non-overlapping partitions, but also overlapping partitions. For non-overlapping partitions, (5.2) is equivalent to (5.1). Considering the generalized modularity at (5.2), we propose a novel formula for evaluation of modularity gain. The proposed formula is simple to implement and computationally efficient as compared to the state-of-the-art [3, 40, 107].

5.3 Fast Fuzzy Modularity Maximization

In this section the FFMM approach is introduced in detail. First, the objective function corresponding to modularity gain attained by changing a vertex membership—a column of the partition matrix $\mathbf{U}$—is proposed in Section 5.3.1. Then, a recursive equation is proposed for updating the partition matrix via maximizing the modularity gain subject to vertex fuzzy membership. The reduced complexity recursive equation for updating the fuzzy partition matrix applicable to large networks is discussed in Section 5.3.2.

5.3.1 Modularity gain objective function

FFMM leverages a simple idea: *update each column of the partition matrix—i.e., vertex fuzzy membership values—such that the modularity value increases at each update.* Given graph $\mathbf{W}$, described in Section 5.2, the modularity gain acquired by updating the nth column of the partition matrix at the $(k-1)$th iteration, from $\mathbf{u}_n^{(k-1)}$ to $\mathbf{u}_n^{(k)}$, is achieved via (see Appendix for proof)

$$\begin{aligned}\Delta Q\left(\mathbf{u}_n^{(k-1)} \rightarrow \mathbf{u}_n^{(k)}\right) = \frac{1}{\sum \mathbf{m}}\Bigg[2\left(\mathbf{u}_n^{(k)} - \mathbf{u}_n^{(k-1)}\right)^T \times \\ \tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n + b_{nn}\left[\left(\mathbf{u}_n^{(k)}\right)^T \mathbf{u}_n^{(k)} - \left(\mathbf{u}_n^{(k-1)}\right)^T \mathbf{u}_n^{(k-1)}\right]\Bigg],\end{aligned} \tag{5.3}$$

where

$$\tilde{\mathbf{U}}_{[n]}^{(k-1)} = \left[\mathbf{u}_1^{(k-1)}, ..., \mathbf{u}_{n-1}^{(k-1)}, \mathbf{0}_{c\times 1}, \mathbf{u}_{n+1}^{(k-1)}, ..., \mathbf{u}_N^{(k-1)}\right] \tag{5.4}$$

and $\mathbf{b}_n$ is the nth column of the modularity matrix $\mathbf{B}$ defined at (5.2). Here, it is aimed to update each column of the partition matrix such that the modularity gain associated with the updated column is maximized. This is achieved by taking the derivative of (5.3) subject to $\mathbf{u}_n^{(k)}$,

$$\frac{\partial\left[\Delta Q\left(\mathbf{u}_n^{(k-1)} \rightarrow \mathbf{u}_n^{(k)}\right)\right]}{\partial\left(\mathbf{u}_n^{(k)}\right)^T} = 2\tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n + b_{nn}\mathbf{u}_n^{(k)} = 0, \tag{5.5}$$

leading to the solution

$$\mathbf{u}_n^{(k)} = \frac{-2\tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n}{b_{nn}}, \quad n = [N]. \tag{5.6}$$

Equation (5.6) is the key equation of FFMM. Considering the fact that the diagonal elements of the adjacency matrix $\mathbf{W}$ are zero, the diagonal elements of the modularity matrix $\mathbf{B} = \left[\mathbf{W} - \frac{\mathbf{m}^T\mathbf{m}}{\|\mathbf{W}\|}\right]$ are negative, i.e., $b_{nn} < 0$ for $n = [N]$. This leads to

$$\mathbf{u}_n^{(k)} = \frac{2\tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n}{|b_{nn}|}, \quad n = [N]. \tag{5.7}$$

The proposed update at (5.7) may lead to negative values. Here, the negative elements are set to zero, then positive elements are normalized to satisfy $\sum_{i=1}^{c} u_{in} = 1$. Applying the normalization on (5.7) enables us to remove $|b_{nn}|$ at (5.7). That simplifies (5.7) into

$$\mathbf{u}_n^{(k)} = \tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n, \quad n = [N], \tag{5.8}$$

where $\tilde{\mathbf{U}}^{(k-1)}$ and $\mathbf{b_n}$ are defined at (5.3). It can be inferred that

$$\mathbf{u}_n^{(k)} = \tilde{\mathbf{U}}_{[n]}^{(k-1)}\mathbf{b}_n = \mathbf{U}^{(k-1)}\tilde{\mathbf{b}}_n, \quad n = [N], \tag{5.9}$$

where $\mathbf{U}^{(k-1)}$ is the partition matrix at the $(k-1)$th iteration and $\tilde{\mathbf{b}}_n$ represents the nth column of modularity matrix B, with zeros at its nth element.

Using (5.9) instead of (5.8) enables constructing the matrix form for updating the

Algorithm 7: Fuzzy Modularity Maximization (small networks)

Require: $\tilde{\mathbf{B}}, C$
1: **return** U
2: Initialize U by uniformly distributed and normalized $C \times N$ matrix
3: **for** $k = 2, 3, ..., K$ **do**
4: $\quad \mathbf{U}^{(k)} = \mathbf{U}^{(k-1)}\tilde{\mathbf{B}}$
5: $\quad$ Eliminate negative elements of $\mathbf{U}^{(k)}$
6: $\quad$ Normalize columns of $\mathbf{U}^{(k)}$ to one
7: **end for**
8: Return $\mathbf{U}^{(k)}$

entire partition matrix simultaneously. The final update equation is

$$\mathbf{U}^{(k)} = \mathbf{U}^{(k-1)}\tilde{\mathbf{B}}, \tag{5.10}$$

where $\tilde{\mathbf{B}}$ represents the modularity matrix $\mathbf{B}$ with an all zeros diagonal.

Algorithm 7 details the proposed FFMM. It performs the overlapping modularity maximization in a single cycle by optimizing the initial (random) partition matrix and then iterating (5.10). However, for large networks ($N \geq 10^4$), computation of $\mathbf{U}\tilde{\mathbf{B}}$ would be complex. In the next section, an efficient technique to reduce the computational complexity associated with calculation of $\mathbf{U}\tilde{\mathbf{B}}$ is discussed.

5.3.2 Efficient computation of $\mathbf{U}\tilde{\mathbf{B}}$

As mentioned in Section 5.3.1, the main computational complexity in Algorithm 7 is associated with $\mathbf{U}\tilde{\mathbf{B}}$. Here, we reduce the computational complexity of $\mathbf{U}\tilde{\mathbf{B}}$ via

a pre-computation of its static components. Using the definition of the modularity matrix $\mathbf{B}$, we have

$$\mathbf{UB} = \mathbf{UW} - \mathbf{U}\frac{\mathbf{m}^T\mathbf{m}}{\sum \mathbf{m}}. \tag{5.11}$$

The nth column of $\mathbf{UB}$ is

$$\mathbf{Ub}_n = \mathbf{Uw}_n - \frac{\mathbf{Um}^T m_n}{\sum \mathbf{m}}, \tag{5.12}$$

where m_n is the nth element of the degree vector $\mathbf{m}$. Here, the term $\mathbf{Um}^T$ is the static component at all columns of $\mathbf{UB}$ and can be pre-calculated for each update of the partition matrix, rather than at each column update. However, (5.10) exploits $\mathbf{U\tilde{B}}$ rather than $\mathbf{UB}$, where $\tilde{\mathbf{B}}$ represents the modularity matrix $\mathbf{B}$ with an all-zero diagonal. Considering that the diagonal elements of the adjacency matrix are zero, the diagonal elements of the modularity matrix can be computed by $b_{nn} = -\frac{m_n^2}{\sum \mathbf{m}}$ for $n = [N]$, where $\mathbf{m}$ is the degree vector and m_n is the nth element of $\mathbf{m}$. This leads to

$$\mathbf{b}_n = \tilde{\mathbf{b}}_\mathbf{n} - \frac{m_n^2 \mathbf{e}_n}{\sum \mathbf{m}}, \tag{5.13}$$

where $\mathbf{e}_n$ denotes an $1 \times N$ vector with $e_i = 1$ for $i = n$ and $e_i = 0$ for $i \neq n$. Substituting (5.13) into (5.12) leads to

$$\mathbf{U}\tilde{\mathbf{b}}_\mathbf{n} = \mathbf{Uw}_n - \frac{\mathbf{Um}^T m_n}{\sum \mathbf{m}} + \mathbf{U}\frac{m_n^2 \mathbf{e}_n}{\sum \mathbf{m}}. \tag{5.14}$$

Applying some mathematical manipulations leads to the final form

$$\mathbf{U}\tilde{\mathbf{b}}_{\mathbf{n}} = \mathbf{U}\mathbf{w}_n + \frac{\left(\mathbf{u}_n m_n - \mathbf{U}m^T\right) m_n}{\sum \mathbf{m}}, \tag{5.15}$$

where the term $\mathbf{U}m^T$ is the static component, as it is independent of n and can be pre-calculated once for all columns of $\mathbf{U}\tilde{\mathbf{B}}$. Moreover, the first component of (5.15) is simplified to

$$\mathbf{U}\mathbf{w}_n = \sum_{l \in \mathbb{N}_n} w_{ln}\mathbf{u}_{ml}, m = [c],\ n = [N], \tag{5.16}$$

where $\mathbb{N}_n$ represents the set of all neighbors of the nth node. Therefore, the final format for calculation of the nth column of $\mathbf{U}\tilde{\mathbf{B}}$ is

$$\mathbf{U}\tilde{\mathbf{b}}_{\mathbf{n}} = \sum_{l \in \mathbb{N}_n} w_{ln}\mathbf{u}_{ml} + \frac{\left(\mathbf{u}_n m_n - \mathbf{U}m^T\right) m_n}{\sum \mathbf{m}}. \tag{5.17}$$

Therefore, the reduced complexity calculation of $\mathbf{U}\tilde{\mathbf{B}}$ incorporates pre-computation of the static part $\mathbf{U}m^T$, then computation of (5.17) for $n = [N]$ to construct all columns of $\mathbf{U}\tilde{\mathbf{B}}$. In the following section multi-cycle FFMM is proposed, which reveals overlapping communities of networks by applying FFMM proposed in Algorithm 7 in detected sub-networks (communities) at each cycle.

5.4 Multi-Cycle FFMM for Large Networks

Multi-cycle FFMM incorporates two modifications applied to the FFMM technique introduced in Algorithm 7. First, direct computation of $\mathbf{U}\tilde{\mathbf{B}}$ is replaced by its reduced complexity version proposed at (5.17). Second, FFMM is applied in multiple cycles. The idea is to use the original FFMM at multiple cycles, where the detected communities at each cycle are considered as the sub-networks to be processed—via FFMM—at the next cycle.

At the first cycle, FFMM is applied to reveal a few super communities, each containing multiple sub-communities. At the next cycle, each super community is considered as a single sub-network and FFMM is applied to detect its communities within. This procedure is repeated to detect communities with higher-and-higher resolutions.

5.4.1 Multi-cyle FFMM

Algorithm 8 details the proposed multi-cycle FFMM for overlapping community detection in large networks.

† **Step 1:** Multi-cycle FFMM starts by applying the FFMM technique to reveal $c^{(1)}$ sub-networks (line 3 of Algorithm 8). Here, FFMM is implemented via

Algorithm 7 and provides the overlapping partition matrix denoted by $\mathbf{U}^{(1)}$.

† **Step 2:** The number of detected sub-networks is set for the first cycle via $C^{(1)} = c^{(1)}$ before starting the second cycle (lines 4 and 5 of Algorithm 8).

† **Step 3:** The sub-network construction algorithm proposed in Section 5.4.2 is applied to the partition matrix derived from previous cycle, i.e. $\mathbf{U}^{(l)}$ (line 6 of Algorithm 8). The sub-network construction algorithm aims to develop the adjacency matrix associated with each detected sub-network.

† **Step 4:** A normalized and randomly distributed partition matrix $\mathbf{U}_j^{(l)} \in \mathbf{R}^{c_j^{(l)} \times N_j^{(l)}}$ is initialized for each constructed sub-network. Here, $N_j^{(l)}$ represents the number of nodes of the jth sub-network at the lth cycle. Moreover, $c_j^{(l)}$ is the number of target communities to be detected in the jth sub-network at the lth cycle. The numbers of target communities for all sub-networks at the lth cycle are derived based on the pre-defined maximum community number associated with that cycle or $c^{(l)}$;

$$c_j^{(l)} = c^{(l)} \frac{N_j^{(l)}}{\underset{i}{max}\left(N_i^{(l)}\right)}, j = [C^{(l-1)}], l \geq 2, \tag{5.18}$$

where j is the index of sub-networks, $c^{(l)}$ is the predefined maximum community

number of the lth cycle, and $N_j^{(l)}$ is the size (i.e., number of vertices) of the jth sub-network at the lth cycle (line 8 of Algorithm 8).

† **Step 5:** Apply FFMM into all available sub-networks to reveal $c_j^{(l)}$ communities for $j = [C^{(l-1)}]$ (line 9 of Algorithm 8).

† **Step 6:** The partition matrix corresponding to the entire network $U^{(l)}$ is constructed, incorporating the proposed technique in Section 5.4.3 (line 11 of Algorithm 8).

† **Step 7:** The number of all detected communities (number of sub-networks for the next cycle) $C^{(l)} = \sum_{j=1}^{C^{(l-1)}} c_j^{(l)}$ (see (5.18) for $c_j^{(l)}$) is calculated and the next cycle starts from **Step 3** (line 12 of Algorithm 8).

5.4.2 Sub-network construction

The main purpose of the sub-network construction process (line 7 in Algorithm 8) is to remove edges among sub-networks. This process is vital as each sub-network is processed individually via FFMM. As shown in Figure 5.1, the input of the sub-network construction process is the network cover matrix $\mathbf{U}^{(l)}$. Then after applying

Algorithm 8: Multi-cycle FFMM

Require: Network Adjacency, $c^{(1)}, c^{(2)}, ..., c^{(L)}$

return $\mathbf{U}^{(L)}$
Initialize $\mathbf{U}^{(1)}$ by $c^{(1)} \times N$ matrix,
Apply FFMM to $\mathbf{U}^{(1)}$ to reveal $c^{(1)}$ communities
Set $C^{(1)} = c^{(1)}$
for $l = 2, ..., L$ **do**
    Construct $C^{(l-1)}$ sub-networks i.e. $\mathbf{W}_j^{(l)}$ for $j = 1, 2, ..., C^{(l-1)}$, from each detected community at the $(l-1)$th cycle as described in Section 5.4.2
    for $j = 1, 2, ..., C^{(l-1)}$ **do**
        Initialize $\mathbf{U}_j^{(l)}$ by $c_j^{(l)} \times N_j^{(l)}$ matrix using (5.18)
        Apply the FFMM to $\mathbf{W}_j^{(l)}$ to detect the $c_j^{(l)}$ communities or $\mathbf{U}_j^{(l)}$.
    end for
    Reform $\mathbf{U}^{(l)}$ incorporating $\mathbf{U}_j^{(l)}$, $j = 1, 2, ..., C^{(l-1)}$ via algorithm described in Section 5.4.3
    Set $C^{(l)} = \sum_{j=1}^{C^{(l-1)}} c_j^{(l)}$
end for

FFMM, the sub-network construction process generates an adjacency matrix for each detected sub-network, i.e., $\mathbf{W}_1^{(1)}$ and $\mathbf{W}_2^{(1)}$ (the adjacency matrices of the black and blue sub-networks in Figure 5.1). In this process each node only keeps those edges that are connected to other nodes of the sub-network. For instance, as shown at the second cycle of Figure 5.1, the node j shares edges with nodes of both detected communities. Therefore, the edges among node j and all other nodes within the second sub-network (i.e. node n) must be removed from the first sub-network, represented by $\mathbf{W}_1^{(1)}$. Likewise, the edges among node j and all other nodes within the first sub-network (i.e. node m) must be removed from the second sub-network, represented by $\mathbf{W}_2^{(1)}$.

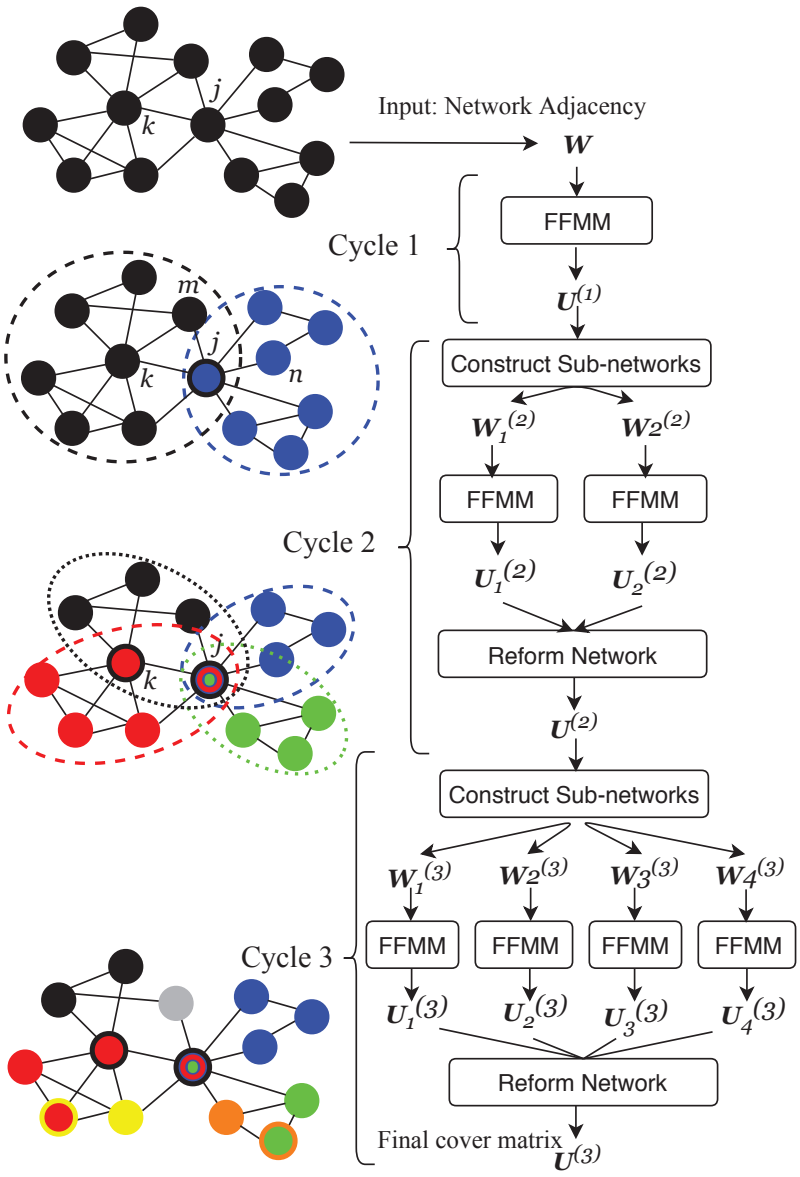


Figure 5.1: Multi-cycle FFMM process in three cycles.

5.4.3 Reform network

The reform network process aims to re-attach the sub-networks by forming the overall network partition matrix. Here, the term overall network partition matrix denotes

the partition matrix associated with the original full network, i.e., $\mathbf{W}$. Merging partition matrices associated with sub-networks is a straightforward process in the case of non-overlapping detected communities, where each node is associated with a single community and, therefore, only one element of each column of the main partition matrix is nonzero. However, FFMM delivers overlapping communities including nodes that are associated into multiple sub-networks in the following cycle; hence, the reforming process is more complicated.

For example, assume $\mathbf{u}_{1,j}^{(1)} = \left[u_{1,1j}^{(1)}, u_{1,2j}^{(1)}\right]^T$ is the membership vector of the jth node at the first cycle. Here, the subscript 1 indicates the index of the sub-network (only one network is available at the first cycle). At the second cycle, FFMM is applied to both detected sub-networks. As shown in Figure 5.1, the first (black) and the second (blue) sub-networks are divided into two other communities. Therefore, at each sub-network a membership vector is associated to the jth node, i.e. $\mathbf{u}_{1,k}^{(2)}$ (the kth column of $\mathbf{U}_1^{(1)}$) and $\mathbf{u}_{2,m}^{(2)}$ (the mth column of $\mathbf{U}_2^{(1)}$), where k and m represent the indices of the jth node in the first and the second sub-networks, respectively. Note that k and m could be the same or different depending on node indexing in each sub-network. The reform network process multiplies the overlapping memberships of the previous cycle into the current cycle membership vectors. This normalizes the jth column of overall network partition matrix to unity. Therefore, at the end of the second cycle the overall membership vector of the jth node corresponding to the four detected communities would be $\mathbf{u}_j^{(2)} = \left[u_{1,1j}^{(1)} \left(\mathbf{u}_{1,k}^{(2)}\right)^T, u_{1,2j}^{(1)} \left(\mathbf{u}_{2,m}^{(2)}\right)^T\right]^T$.

In general, if node n is associated with communities i_1 and i_2 within the jth sub-network at the $(l-1)$th cycle, then $u_{j,i_1n}^{(l-1)} \neq 0$ and $u_{j,i_2n}^{(l-1)} \neq 0$. At the lth cycle, the nth column of the overall partition matrix $\mathbf{u}_n^{(l)}$ is

$$\mathbf{u}_n^{(l)} = \left[0, ..., 0, u_{j,i_1n}^{(l-1)} \left(\mathbf{u}_{i_1,k}^{(l)}\right)^T, 0, ..., 0, u_{j,i_2n}^{(l-1)} \left(\mathbf{u}_{i_2,m}^{(l)}\right)^T, 0, ..., 0\right]^T \tag{5.19}$$

where $\mathbf{u}_{i_1,k}^{(l)}$ and $\mathbf{u}_{i_2,m}^{(l)}$ represent the kth and the mth column of the partition matrices $\mathbf{U}_{i_1}^{(l)}$ and $\mathbf{U}_{i_2}^{(l)}$, respectively. Moreover, m and k represent the indices of the nth node at the i_1 and i_2 sub-networks at the lth cycle, respectively.

5.4.4 Non-Overlapping membership convergence

The fuzzy nature of FFMM produces large numbers of overlapping nodes with membership values close to non-overlapping (crisp) values of 1 or 0. These memberships converge to the non-overlapping values by increasing the number of iterations (K in Algorithm 7); however, they can be forced to the closest crisp value after passing a pre-defined threshold, denoted as τ. Therefore, at the lth cycle (where $l \neq L$), nodes with membership values satisfying $\left(u_{kn}^{(l)} - \frac{1}{C^{(l)}} \sum_i u_{in}\right) > \tau$ will be forced to non-overlapping membership in the community with the highest membership value. Here, τ is a predefined value selected as 0.1 in all simulations. This approach guarantees more fuzzy memberships of vertices with higher community ambiguity (lower standard

deviation for membership value) or betweenness, and non-overlapping (crisp) membership values for nodes with lower community ambiguity (higher standard deviation for membership value).

5.5 Experiments

Experiments were conducted to verify the effectiveness and efficiency of the proposed technique. Here, the proposed method is evaluated by analyzing small, medium-size, and large real-world data sets. Moreover, the undirected and unweighted benchmark network called *Lancichinetti-Fortunato-Radicchi* (LFR) [115, 116] is selected to evaluate performance of the proposed technique for a network with known overlapping community structure. Selected performance metrics include Newman's modularity value, overall computation time, and the number of fuzzy nodes for real world data sets. The number of fuzzy nodes in the final community structure is important for real-world analysis of the communities. Fuzzy nodes often are used to find community members that are between communities, such as important conduits in criminal networks or people that are advantageous to advertise to in word-of-mouth marketing campaigns. Hence, there is a trade-off between the modularity value of the communities found and the number of fuzzy nodes discovered in those communities. Thus, we wish to find high-modularity community structure while maximizing the number of fuzzy nodes. Moreover, *Overlapping Normalized Mutual Information* (ONMI) [136]

is evaluated for the benchmark LFR networks. The parameters used in the experiments are discussed in detail next to ease reproduction of our results. Then, then performance of our proposed multi-cycle FFMM is evaluated against state-of-the-art techniques in Section 5.5.2.

5.5.1 Experiment parameters

Table 5.2 contains the details of the data sets we used and their corresponding algorithm parameters. The third column of Table 5.2 indicates applied resolution at each cycle, $[c^{(1)}, c^{(2)}, ..., c^{(L)}]$. As shown in Table 5.2, FFMM (single-cycle FFMM) is applied to small networks such as Dolphin through Email; meanwhile, for larger networks, multi-cycle FFMM is used. The fourth column of Table 5.2 indicates the number of iterations K of Alg. 8. We also measure the computation time of multi-cycle FFMM and compare to the state-of-the-art. Here, the experiments are run on the same machine to develop a fair comparison. The experiment was conducted on a laptop with an i7-6560U processor @2.20GHz with 16GB of memory and all software was coded in Matlab.

Table 5.3 describes selected parameters for the LFR synthetic networks. The LFR benchmark generates networks that contain common features of real-world data sets by assigning various values to the parameters in the Table 5.3. Here, N denotes the

Table 5.2
Applied simulation parameters

Network	N	$[c^{(1)}, c^{(2)}, ..., c^{(L)}]$	K	Type
Dolphin [88]	62	5	50	Small
Football [118]	115	10	50	Small
Jazz [90]	198	4	50	Small
Metabolic [27]	453	9	50	Small
Email [119]	1,133	11	50	Small
Facebook [120]	4,039	15	50	Small
Email-Enron [121]	36,692	10,100	50	Large
Com-dlbp [121]	317,080	20,100	75	Huge
Com-youtube [121]	1,134,890	20,100	75	Huge
LFR1 [115]	1e3	100	50	Small
LFR2 [115]	1e4	10,100	50	Large
LFR3 [115]	1e5	20,100	50	Large

number of nodes; C_{avg} is the average number of nodes within network communities; C_{max} is the number of nodes within the largest community; k_{avg} is the average degree of the nodes; k_{max} is the maximum degree of the nodes in the network. Moreover, μ represent the mixing parameter in the LFR network. The mixing parameter denotes the average rate of edges that connect nodes from different communities. Therefore, it is expected that by increasing the value of μ the strength of the community structure decreases and makes the community detection more difficult. Some parameters such as $\tau = 1$ (the exponent for the degree sequence) and $\beta = 2$ (the exponent for the community size distribution), are set to the same value for all generated benchmark LFR networks.

Table 5.3
Parameters of LFR networks

Net.	N	C_{avg}	C_{max}	k_{avg}	k_{max}	μ
LFR1	1e3	50	100	25	100	$[0.1 - 0.9]$
LFR2	1e4	100	200	50	200	$[0.1 - 0.9]$
LFR3	1e5	200	500	50	200	$[0.1 - 0.9]$

5.5.2 Experiment results

Tables 5.5 and 5.6 present performance of multi-cycle FFMM compared to state-of-the-art overlapping community detection approaches, such as Multi-cut Spectral FCM (H2) [125] and Fuzzy Agglomerative Community Detection (FuzAg) [134], for a variety of benchmark real-word networks [88, 90, 118, 119, 120, 121]. It is observed that FFMM outperforms H2 in terms of modularity value with a much lower execution time. The FuzAg approach produces slightly better modularity values; however, the associated time complexity is way too high to handle networks larger than the Email data set.

Table 5.7 presents performance of multi-cycle FFMM compared to non-overlapping Louvain [3]. Here, the Louvain approach is selected to compare the computational complexity associated with both techniques. It should be noted that the non-overlapping nature of detected communities from (non-overlapping) approaches such as Louvain leads to higher Newman's modularity values as compared to fuzzy approaches such as multi-cycle FFMM, H2 [125], or FuzAg [134]. However, it is observed

Table 5.4
Performance analysis of time complexity comparison of proposed FMMM with FuzAg ,FMM/H2, NGTCDA

Algorithm	FMMM	FuzAg	FMM/H2	NGTCDA
Time Complexity	$O(N)$	$O(N^2)$	$O(N^2)$	$O(N^2)$

that multi-cycle FFMM presents modularity values comparable with the Louvain approach at lower computational time. In [130] the *new game-theoretic community detection algorithm* (NGTCDA) is proposed as a fuzzy community detection technique using modularity maximization. In [130] the authors obtained 0.7708/0.2708 and 0.6417/0.0239 values for maximum/final modularity values at 67 and 364 minutes for Facebook and Email-Enron networks, respectively. Comparing these results, it can be observed that multi-cycle FFMM obtains higher modularity values with higher numbers of fuzzy nodes at a running time about a *thousand times faster*. We also use non-overlapping Louvain heuristic method [3] for compression with proposed overlapping FFMM for the same data-sets. It should be emphasized that the Louvain approach [3] is one of best performing non-overlapping modularity-based approaches for large data set to date; most recent modularity-based overlapping methods, such as NGTCDA [130], have poor performance in term of computational time and final modularity value. Furthermore, other overlapping approaches can be difficult or impossible to apply to large data sets, such as H2 [125] and FuzAg [134], due to computational complexity. To our knowledge, multi-cycle FFMM has, by far, the best performance and fastest execution time of all overlapping community detection approaches.

Table 5.5
Simulation results over 100 runs: overlapping FFMM versus overlapping FuzAg

Network	FuzAg [134]			(multi-cycle) FFMM		
Parameter	Q Avg.	t (sec) Avg.	CAvg.	Q Avg.	t (sec) Avg.	C Avg.
Dolphin	0.6642	323	7	0.5285	0.002	5
Football	0.6915	715	14	0.592	0.004	10
Jazz	0.4624	3,381	6	0.440	0.005	4
Email	0.5681	28,123	14	0.560	0.063	10
Facebook	-	-	-	0.825	2.65	15
Email-Enron	-	-	-	0.4505	17.2	694.2
Com-dlbp	-	-	-	0.7173	305.3	1,974.6
Com-youtube	-	-	-	0.5254	685.9	622.3

Table 5.6
Simulation results over 100 runs: overlapping FFMM versus overlapping H2

Network	H2 [125]			(multi-cycle) FFMM		
Parameter	Q Avg.	t (sec) Avg.	C Avg.	Q Avg.	t (sec) Avg.	C Avg.
Dolphin	0.5285	35	5	0.500	0.002	5
Football	0.6046	77	10	0.592	0.004	10
Jazz	0.4452	135	4	0.440	0.005	4
Email	0.5491	8221	9	0.560	0.063	10
Facebook	-	-	-	0.825	2.65	15
Email-Enron	-	-	-	0.4505	17.2	694.2
Com-dlbp	-	-	-	0.7173	305.3	1,974.6
Com-youtube	-	-	-	0.5254	685.9	622.3

Table 5.8 presents the number of overlapping nodes detected by FFMM. Figure 5.2 plots ONMI versus mixing parameter for the benchmark network LFR1, described in Table 5.3. Here, various number of overlapping nodes N_{ov} are incorporated for (a) $c_{ov} = 2$ and (b) $c_{ov} = 5$, where c_{ov} denotes the number of associated overlapping communities for each node. For this experiment, single-cycle FFMM was used to detect communities. As shown, FFMM performs very well (ONMI > 0.9) in terms of ONMI for $N_{ov} = 10, 20, 50$ at $c_{ov} = 2$ for a wide range of mixing parameter values.

Table 5.7
Simulation results over 100 runs: fuzzy FFMM versus non-overlapping louvain

Network	Louvain [3]			(multi-cycle) FFMM		
Parameter	Q Avg.	t (sec) Avg.	C	Q Avg.	t (sec) Avg.	C Avg.
Dolphin	0.519	0.102	5	0.500	0.002	5
Football	0.604	0.105	9	0.592	0.004	10
Jazz	0.443	0.152	3	0.440	0.005	4
Email	0.540	0.859	11	0.560	0.063	10
Facebook	0.8323	4.06	15.60	0.825	2.65	15
Email-Enron	0.5845	54.58	1185.3	0.4505	17.2	694.2
Com-dlbp	0.8102	1086.8	149.4	0.7173	305.3	1,974.6
Com-youtube	0.712	1037.7	–	0.5254	685.9	622.3

Table 5.8
Simulation results over 100 runs: Number of overlapping nodes

Network	Multi-cycle FFMM
Parameter	N_o Avg.
Dolphin	4.2
Football	2.3
Jazz	11.4
Email	60.7
Facebook	104
Email-Enron	381.3
Com-dlbp	2055.4
Com-youtube	5083.8

The same results are observed for LFR networks with $c_{ov} = 5$ for $N_{ov} = 10$and20. Although, it is noted that increasing the number of overlapping nodes N_{ov} degrades the overlapping community detection process. But more overlapping nodes means less overall community structure; hence, this result is expected.

Figure 5.3 shows ONMI versus mixing parameter for the benchmark network LFR2 described in Table 5.3. Here, various numbers of overlapping nodes N_{ov} are incorporated for (a) $c_{ov} = 2$ and (b) $c_{ov} = 5$. The multi-cycle FFMM approach proposed in

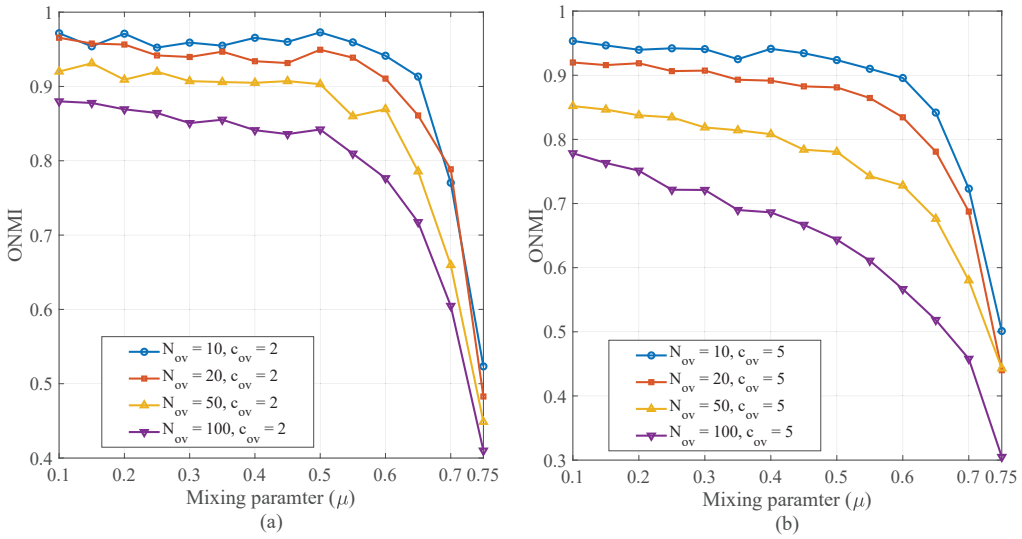


Figure 5.2: Average ONMI for benchmark network LFR1 with various numbers of overlapping nodes.

Algorithm 8 was used to detect communities. As shown, FFMM performs very well (ONMI > 0.9) in terms of ONMI for $N_{ov} = 50, 100, 200\text{and}500$ at $c_{ov} = 2$ for a wide range of mixing parameter values. The same results are observed for LFR networks with $c_{ov} = 5$ for $N_{ov} = 50, 100 and 200$.

Figure 5.4 shows ONMI versus mixing parameter for the benchmark network LFR3, described in Table 5.3. Here, various number of overlapping nodes N_{ov} are incorporated for (a) $c_{ov} = 2$ and (b) $c_{ov} = 5$. Multi-cycle FFMM performs very well (ONMI > 0.9) for all scenarios. Moreover, it is observed that due to increasing mixing parameter, the numbers of overlapping nodes N_{ov}, i.e., the numbers of overlapping communities per node c_{ov}, increases. Subsequently, ONMI decreases, which is expected.

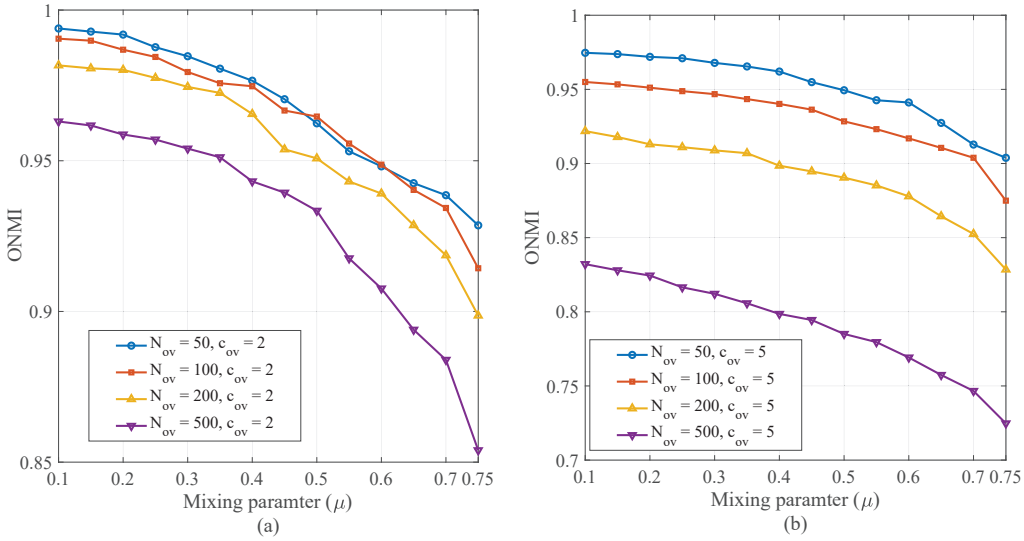


Figure 5.3: Average ONMI for benchmark network LFR2 with various number of overlapping nodes.

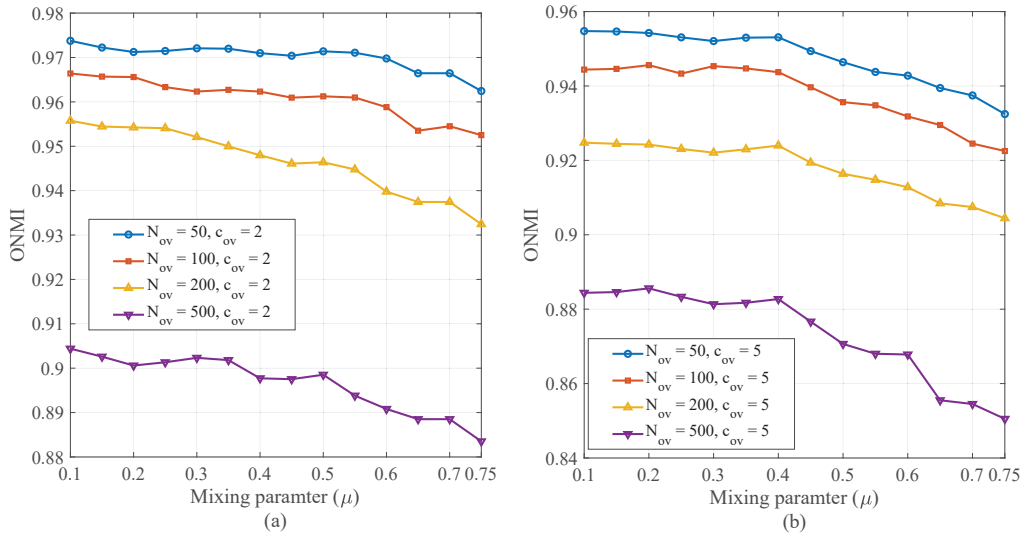


Figure 5.4: Average ONMI for benchmark network LFR3 with various number of overlapping nodes.

Figure 5.5 visualizes four detected communities found by FFMM and corresponding

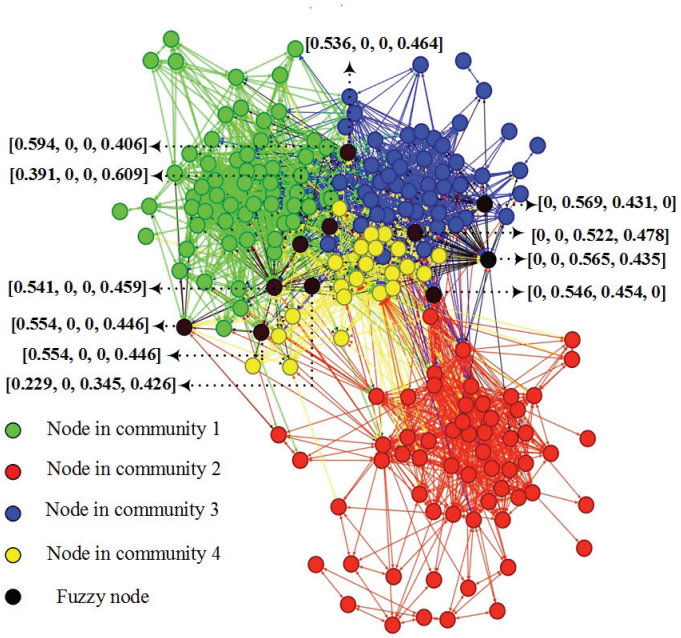


Figure 5.5: Detected communities of Jazz network and membership values of fuzzy nodes.

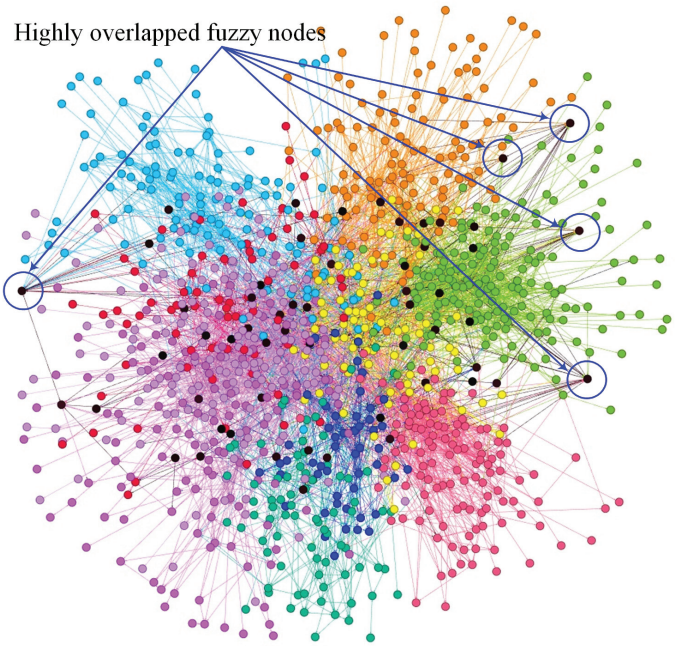


Figure 5.6: Detected communities of Email network.

membership values of fuzzy nodes in the Jazz network [90]. As shown, fuzzy membership values are allocated to nodes with high betweenness and lower centrality. Moreover, the second community, which shares fewer connections with other communities, includes only 2 fuzzy values; these are lower than 10% of all network fuzzy membership values. Figure 5.6 depicts the community detection found by FFMM for the Email network [119]. Here, 10 communities are detected, which are visualized in various colors. Nodes with fuzzy membership are highlighted in black to stress their contribution in various communities. Moreover, several fuzzy nodes with high betweenness are pulled out to show the effectiveness of FFMM technique in detecting highly overlapping nodes. The visualized networks in Figs. 5.5 and 5.6 are examples of real world networks indicating efficiency of FFMM to segregate network communities, while highly overlapped nodes share multiple communities.

Summary of Multi-Cycle FFMM In this study, an approach for overlapping community detection was introduced called FFMM. The FFMM method uses a recursive equation derived from optimization of the modularity change associated with changing a column of the partition matrix. Then, multi-cycle FFMM was proposed as a feasible solution for community detection in large networks. Our results over real-world data sets and benchmark networks demonstrated the efficiency of multi-cycle FFMM in terms of modularity, computational complexity, and overlapping NMI, as

compared to existing solutions. Moreover, near linear computational time of multi-cycle FFMM introduced a new breakthrough for fuzzy community detection for large and huge networks.

Chapter 6

Range-Free Anchor Selection in Wireless Sensor Networks via Community Detection

The material in this chapter is in preparation for submission to IEEE Transactions on Computational Social Systems.

6.1 Introduction

Range measurements such as time of arrival (ToA) [137] are vital for *Wireless Sensor Network* (WSN) localization. On average, more range measurements leads to higher location accuracy, but more measurements incur higher energy consumption. In distributed localization, range measurements are applied between the target nodes (i.e., nodes with unknown locations) and anchor nodes (i.e., nodes with known locations). In a dense WSN, such as in smart cities, large buildings, airports, etc., target nodes are able to find large numbers of anchor nodes in their vicinity. Although applying range measurements with all anchor nodes may lead to higher location accuracy, this increases energy consumption and network traffic load. Because of this, there is much research on anchor selection approaches in WSNs. A few works studied anchor selection in range-free localization [138, 139, 140]. Some other works have discussed the anchor selection approaches for range-based localization [141, 142, 143, 144, 145, 146, 147]. In [141], localization is conducted by selecting anchors with the strongest received signals strength. However, stronger received signal strength indicator (RSSI) or closer range does not guarantee the best anchors selection for localization accuracy. Authors of [145] and [147] exploited range to select anchors with the smallest measured distance. Moreover, in [146] both measured ranges and also the information about their coarse variances were exploited for best anchor selection. However, the use of range for anchor selection increases ranging

cost and does not guarantee ambiguity-free and accurate localization.

In this chapter, both non-overlapping and overlapping memberships of target nodes and anchor nodes are utilized for anchor selection for distributed localization of dense WSNs. Two objective functions, based on non-overlapping and overlapping detected communities, are proposed. The proposed objective functions exploit the community membership of each node and its neighbors. Here, classic *label propagation* (LP) [2], HLP discussed in chapter 3, and Louvain [148] community detection methods are utilized for non-overlapping community detection to detect nodes' crisp community memberships. FFMM and Multi-Cycle FFMM, discussed in chapter 5, are utilized to detect nodes' fuzzy community memberships.

6.2 Range-Free Anchor Selection via Community Detection

This section discusses in detail the proposed range-free anchor selection approach via community detection. Here, it is assumed that all sensor nodes in the network (including anchor nodes) can communicate directly with other nodes/anchors in their vicinity and can route data into a processing center or hub. The nth node/anchor is considered as the neighbor of the mth node if they can detect the beacon associated with each other. It is assumed that each node transmits a unique ID which

differentiates it from all other nodes within the network.

Once each node reveals available nodes/anchors in their vicinity, they communicate the detected IDs received into the processing center. The processing center constructs a network based on this connection information and runs a community detection approach to associate a overlapping or non-overlapping community membership to each node. The processing center then executes the proposed anchor selection algorithm to select the optimum anchor set among all available combinations, and routes back the results (IDs of selected anchor nodes) to each sensor node. Once the sensor nodes receive the selected anchors' IDs, the ranging process is initialized. Then, each node would be able to run distributed localization using its associated range measurements.

In the following sections, the objective function corresponding to both non-overlapping and also overlapping community detection is studied in detail.

6.2.1 Anchor selection using non-overlapping community detection

The range-free anchor selection approach is developed based on maximizing the community distance (minimizing community membership coherence) among selected anchors while the community distance is minimized (maximizing community membership coherence) among selected anchors and the target nodes. This approach is developed based on the fact that the probability of choosing anchors in only one geometric area (which leads to location ambiguity) decreases when they are selected from different communities. Their community distance are minimized with the target node so that each target node has nearby anchors. Hence, the algorithm seeks to maximize the overall coverage of the anchors, while maintaining a dense network of anchors to improve localization accuracy.

Consider S_k as the candidate set of anchor nodes within the vicinity of the kth node. The proposed range-free anchor node selection objective function is

$$\begin{aligned} S_k &= \underset{n}{argmax} \left\{ \sum_{i \in \mathbf{a}_{k,n}} \sum_{j \in \mathbf{a}_{k,n}, j \neq i} (1 - \delta_l(i,j)) - \gamma \sum_{j \in \mathbf{a}_{k,n}} (1 - \delta_l(i,j)) \right\} \\ &= \underset{n}{argmin} \left\{ \sum_{i \in \mathbf{a}_{k,n}} \sum_{j \in \mathbf{a}_{k,n}, j \neq i} \delta_l(i,j) - \gamma \sum_{j \in \mathbf{a}_{k,n}} \delta_l(k,j) \right\} \end{aligned} \tag{6.1}$$

where $\mathbf{a}_{k,n}$ denotes the nth column of the anchors permutation matrix in the vicinity

of the kth node, i.e., $\mathbf{A}_k$. Moreover, γ and $\delta_l(i,j)$ represent the weight coefficient and the Dirac delta function of nodes' community memberships, where $\delta_l(i,j) = 1$ when the ith and the jth anchor are within the same community (i.e. $l_i = l_j$), and $\delta_l(i,j) = 0$ elsewhere (i.e. $l_i \neq l_j$). Here, the weight coefficient γ aims to increase impact of the overall community distance among all members of the candidate sub-set of anchors (first term of proposed object function) to guarantee each individual anchor of candidate sub-set is belong to a different community, over to the community distance of each anchor of that sub-set of anchor nodes with respect to the target node (second term of proposed object function).

Using (6.1) may lead to multiple solutions due to the same overall value of objective function. In such cases, an additional range-free objective function is added to select the best solution between candidates associated with (6.1);

$$S_k = \underset{m}{argmax} \left\{ \sum_{i \in \tilde{\mathbf{a}}_{k,m}} \sum_{j \in \tilde{\mathbf{a}}_{k,m}} \|X_i - X_j\|^2 \right\} \tag{6.2}$$

where $\tilde{\mathbf{a}}_{k,m}$ represents the set of solutions obtained by (6.1) (if more than one solution is derived) and X_i are the coordinates of the ith anchor node.

6.2.2 Anchor selection using overlapping community detection

When overlapping community detection, such as FFMM, is used, the objective function of anchor selection aims to maximize the community distance among the anchors and minimize the community distance among anchors and the target nodes. However, the membership value in non-overlapping community detection is represented by a c vector, where c represents the number of communities. Therefore, the candidate set of anchor nodes in the vicinity of the kth node (S_k) can be computed by

$$S_k = \underset{n}{argmax} \left\{ \sum_{i \in \mathbf{a}_{k,n}} \sum_{j \in \mathbf{a}_{k,n}, j \neq i} \|\mathbf{u}_i - \mathbf{u}_j\|_1 - \gamma \sum_{j \in \mathbf{a}_{k,n}} \|\mathbf{u}_k - \mathbf{u}_j\|_1 \right\} \tag{6.3}$$

where $\mathbf{a}_{k,n}$ is defined at (6.1) and $\mathbf{u}_i$ represents the fuzzy membership values of the ith node or the ith column of the detected community cover matrix $\mathbf{U}$. Moreover the $\|.\|_1$ notation represents the ℓ_1-norm of a vector.

It should be noted that by using overlapping community detection approaches, such as FFMM, some nodes (usually more than 50%) would have non-overlapping community memberships (i.e., not 0 or 1). Therefore, it is probable that all of the anchors within a selected set have non-overlapping membership, which may lead to more than one solution to (6.3). We believe that the probability of such cases is more slim than in the

Table 6.1
Network parameters

Network	N	N_a	Dimensions	Type
Net. I	500	200	[-500,500]	Random node and anchor
Net. II	500	15^2	[-500,500]	Random node fixed anchor
Net. III	2000	500	[-1000,1000]	Random node and anchor
Net. IV	2000	25^2	[-1000,1000]	Random node fixed anchor
Net. V	10000	500	[-1000,1000]	Random node and anchor
Net. VI	10000	25^2	[-1000,1000]	Random node fixed anchor

use of the non-overlapping objective (6.1); however, for such cases the proposed sub-objective at (6.2) is used to select the final solution between the (possible) multiple candidates obtained by (6.3).

6.3 Experimental Results and Discussion

Simulations are conducted to study the performance of the proposed range-free anchor selection approach for a variety of randomly generated dense WSNs. Here, CVX tools [149] are used to simulate the distributed localization via a *semi-definite program* (SDP) solver. Table 6.1 contains the parameters used for six different simulated networks. The parameters N and N_a represent the number of target and anchor nodes, respectively.

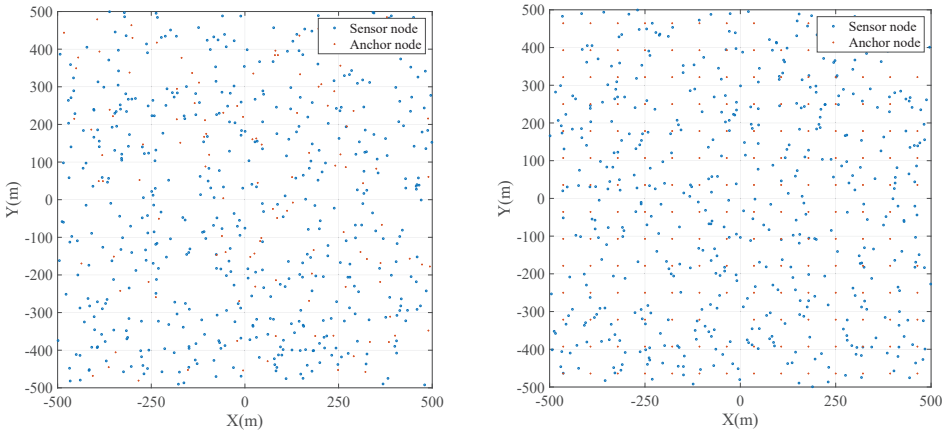
6.3.1 Simulation parameters and methods

As shown in Table 6.1, two different types of networks are considered for simulation. For both types, target nodes are randomly distributed using uniform distributions. Networks with random target and anchor nodes aim to simulate WSNs with ad-hoc properties, where most of anchor nodes are actually part of network and may change their locations over time. However, networks with random target nodes and equally-spaced anchors aims to simulate distributed WSNs in environments such as smart cities or airports, where supervised anchor node placement is practical. Figure 6.1 depicts examples of the two network types with random and equally-spaced anchor distribution. Table 6.2 contains the parameters of community detection approaches used, including LP, HLP and (Multi-Cycle) FFMM. The average normalized location error is selected as the performance benchmark, which is computed as follows,

$$Error = \frac{1}{N}\sum_{k=1}^{N}\frac{\left\|X_k - \hat{X}_k\right\|^2}{\|X_k\|^2}. \tag{6.4}$$

Table 6.2
Simulation parameters applied for community detection approaches

Approach	(H)LP		(Multi-Cycle) FFMM		
Network	c	K	$[c^{(1)}, c^{(2)}]$	K	τ
Net. I	10	10	10	50	0.2
Net. II	10	10	10	50	0.2
Net. III	20	10	$[5, 10]$	50	0.2
Net. IV	20	10	$[5, 10]$	50	0.2
Net. V	20	10	$[10, 10]$	50	0.2
Net. VI	20	10	$[10, 10]$	50	0.2

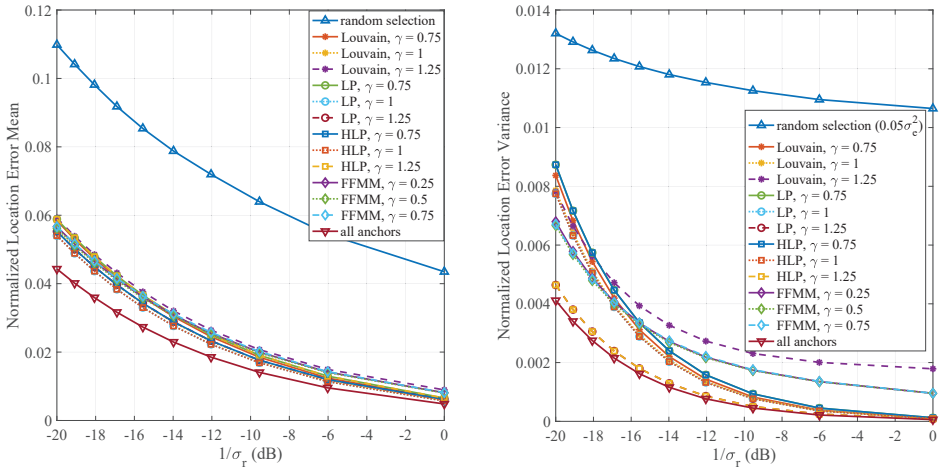


(a) Random anchor distribution (b) Equally spaced anchor distribution

Figure 6.1: Visualization of synthetic Networks I and II, (a) nodes and anchors are randomly distributed via uniform distribution, (b) nodes are randomly distributed via uniform distribution where anchors are equally spaced.

6.3.2 Results and discussions

Figures 6.2(a,b) depict the average and variance of normalized error corresponding to Network 1 using both non-overlapping and overlapping community detection approaches. As shown, localization error is compared with a random selection of a set of



(a) Mean (b) Variance

Figure 6.2: (a) Mean and (b) Variance of normalized localization error using the proposed range-free anchor selection over Net I.

$K_a = 3$ anchors and also to using all anchors, i.e., full anchor measurements. Simulation results demonstrate that the proposed technique is quite effective in terms of both mean and variance of location error. However, it is observed that non-overlapping communities via LP and HLP produce slightly better results, especially for lower range measurement noise levels.

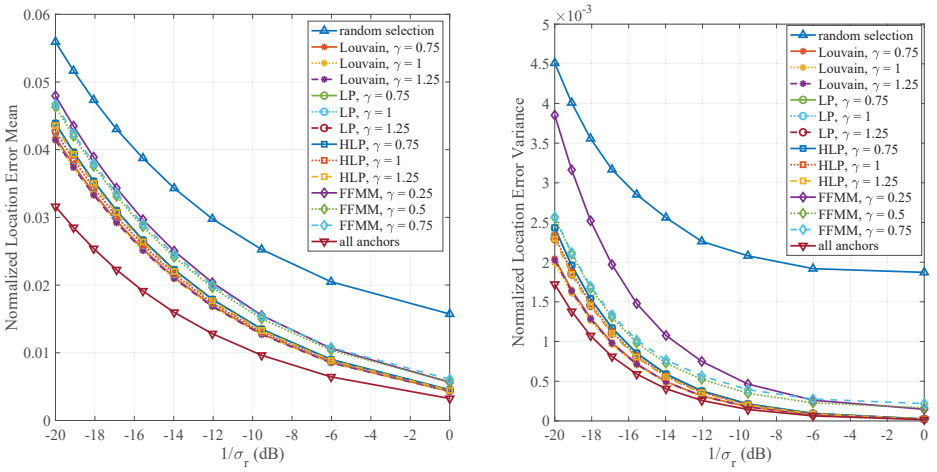
Here, various weight coefficients (γ) are evaluated for both objective functions: (6.1) and (6.3). Simulation results show that the performance of all selected coefficients are nearly the same; however, the best results are obtained with $\gamma = 1$ with the non-overlapping objective, and and $\gamma = 0.5$ with the overlapping objective.

Figures 6.3(a,b) depict the mean and variance of normalized location error corresponding to Network 2. The difference between of the performance of the random

sub-set anchors selection and all anchors measurements observed in Figures 6.2 (a) and (b). However, by using the proposed anchor selection technique illustrate significantly performance improvements (or decreasing order of normalized location error mean and variance) specifically at lower range measurement noise.

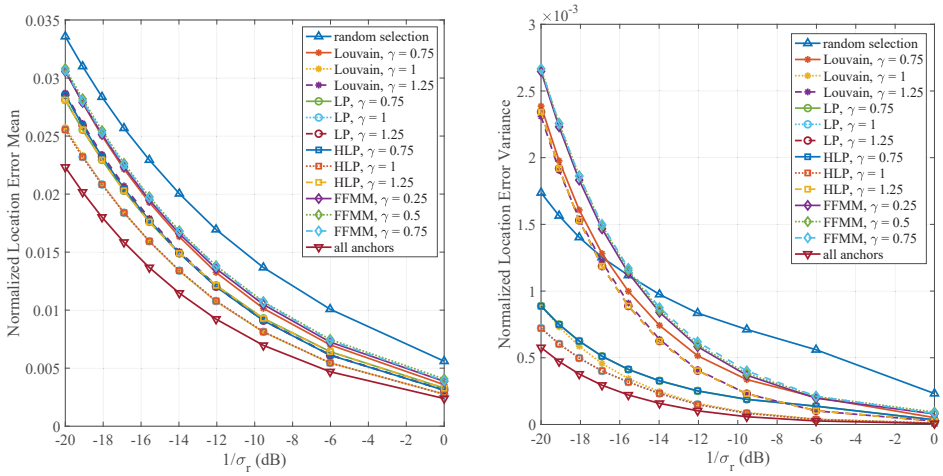
Figures 6.4, 6.5, 6.6 and 6.7 depict the average normalized error corresponding to Networks 3–6, respectively. It is observed that by increasing the number of nodes the difference of performance between the random selection and full anchor measurements shrinks. Meanwhile, the proposed technique performs very well, especially at lower range measurement noise levels. This is due to the fact that the proposed technique aims to minimize the error imposed by location ambiguity, which is not the major component of error at noisy range measurements. However, at lower range measurement noise, the localization error imposed by range measurements is not the main driver of error; instead, the location ambiguity is the major part of the average localization error. Therefore, the proposed technique has much more influence at lower range measurement noise levels, where the major error—location ambiguity—is addressed via the proposed approach.

Summary of Range-Free Anchor Selection via Community Detection In this study, a variety of community detection approaches were applied to the localization problem in WSNs. As a novel application of community detection, a range-free anchor selection method was developed that can utilize both overlapping and also



(a) Mean (b) Variance

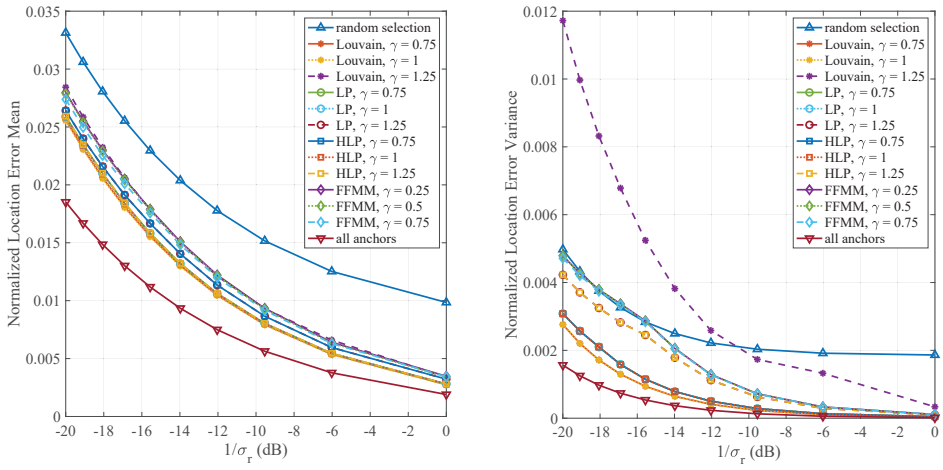
Figure 6.3: (a) Mean and (b) Variance of normalized localization error using the proposed range-free anchor selection over Net II.



(a) Mean (b) Variance

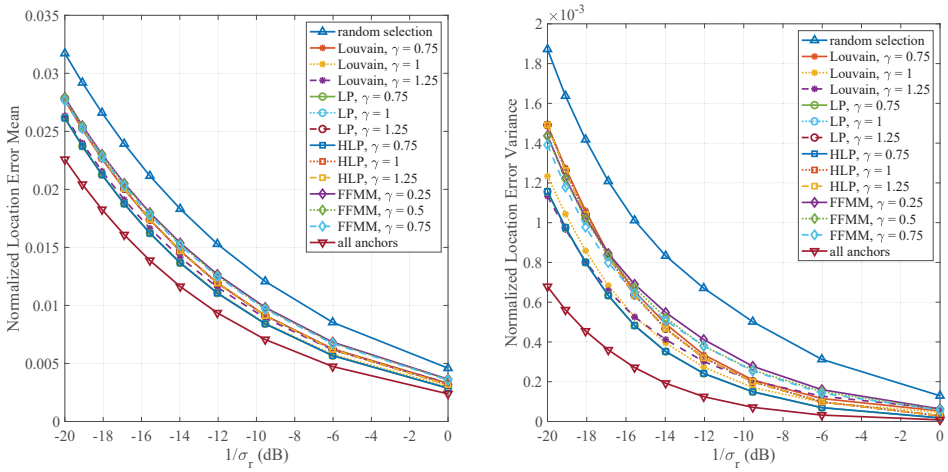
Figure 6.4: (a) Mean and (b) Variance of normalized localization error using the proposed range-free anchor selection over Net III.

non-overlapping community detection. The proposed technique exploited a proposed community membership distance objective function to select the best anchor nodes



(a) Mean (b) Variance

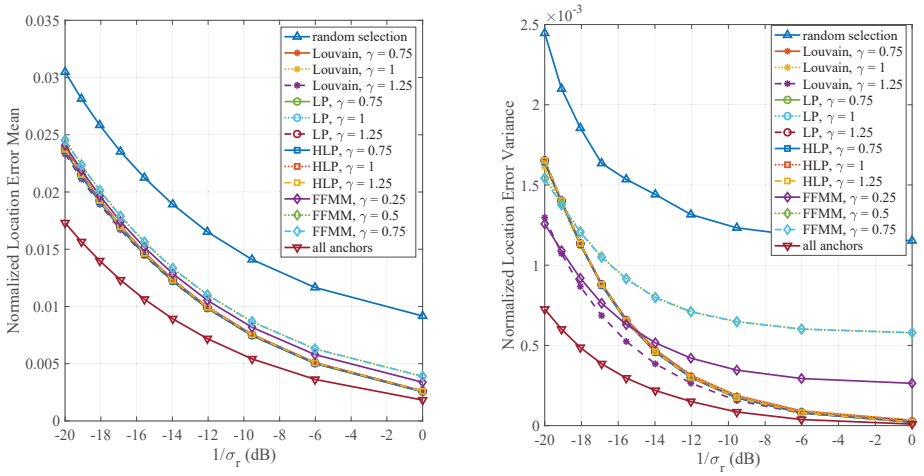
Figure 6.5: (a) Mean and (b) Variance of normalized localization error using the proposed range-free anchor selection over Net IV.



(a) Mean (b) Variance

Figure 6.6: (a) Mean and (b) Variance of normalized localization error using the proposed range-free anchor selection over Net V.

among a set of candidate nodes in a network. Simulation results over a variety of networks demonstrated that the proposed technique is efficient and effective, especially in networks with low relative range measurement error where location ambiguity is



(a) Mean (b) Variance

Figure 6.7: (a) Mean and (b) Variance of normalized localization error using the proposed range-free anchor selection over Net VI.

the largest driver of localization error.

Chapter 7

Conclusions

Community detection is an important problem in complex network analysis. Community detection and graph clustering are categorized as NP-complete problems with no globally optimal solution. Most community detection approach found in the literature suffer from high computational complexity throughout the detection process or low performance in terms of modularity, NMI or ONMI. This research targeted the development of non-overlapping and overlapping community detection algorithms which are capable of producing high quality detection accuracy while minimizing time consumption throughout the detection process. Also, in this work it is aimed to develop a novel technique for anchor selection in very dense WSNs. We propose using both overlapping and non-overlapping community detection approaches to develop range-free approach for optimum anchor selection. The following items are summary of our research on the topics addressed in this dissertation.

Completely Positive Programming In this study, a novel method for *Modularity Maximization* (MM) for community detection is presented which uses the *Alternating Direction Augmented Lagrangian* (ADAL) method for maximizing a generalized form of Newman's modularity function. We first transform Newman's modularity function into a quadratic program and then use *Completely Positive Programming* (CPP) to map the quadratic program to a linear program, which provides the globally optimal maximum modularity partition. In order to solve the proposed CPP problem, a closed form solution using the ADAL merged with a rank minimization approach is

proposed. The performance of the proposed method is evaluated on several real-world data sets used for benchmarks community detection. Simulation results shows the proposed technique provides outstanding results in terms of modularity value for crisp partitions.

Hybrid Label Propagation In this study, community detection via a novel hybrid label propagation approach was proposed, which utilizes a novel *hybrid label propagation* (HLP) approach for maximizing a generalized form of Newman's modularity function. The proposed technique leverages an efficient approach to select the best label transition. Here, a novel objective function is developed to maximize the modularity variation corresponding to each label propagation. Moreover, a hybrid form of synchronous and asynchronous label propagation is developed by exploiting dynamic and static label lists. The static label list is utilized for pre-calculation of static components associated with the modularity variation objective function; meanwhile, the dynamic label list is used to update components with lower computational complexity at each iteration. Efficiency, stability, and scalability of the proposed technique are investigated for both synthetic graphs and empirical data sets and compared with state-of-the art techniques in terms of modularity, normalized mutual information (NMI), and computational complexity. Simulation and real-world network results prove the efficiency of the proposed approach which produces competitive modularity and NMI values at a lower computational complexity, specifically for large networks.

Linear Time Modularity Gain Acceleration In this study, we introduced a novel objective function for calculation of the attained modularity gain corresponding to label transitions. Here, modularity gain calculations are replaced with an objective function that quantifies available label transitions; we call our approach the Modularity Gain Acceleration (MGA) approach. The proposed technique is simplified and divided into two components, the local and global sum-weights. The Local Sum-Weight (LSW) is the component with lower complexity and is calculated for each candidate label transition. However, the General Sum-Weight (GSW) is more computationally complex, and is calculated only once per each label. GSW is updated by leveraging a simple process for each node-label transition, instead of for all available labels. The proposed technique is applied to selected state-of-the-art LP-based community detection methods and the resulting network modularity and execution time are compared with traditional methods. By applying MGA to LP-based methods, the run-time is significantly reduced—sometimes finishing before the traditional approach even finishes one iteration—and the same modularity result and number of communities, i.e., community detection result, is obtained. The MGA approach leads to significant efficiency improvements by reducing time consumption up to 85% relative to the original algorithms with the exact same quality in terms of modularity value.

Multi-Cycle FFMM In this study, an approach for overlapping community detection was introduced called FFMM. FFMM uses novel iterative equations to calculate the gain associated with changing the fuzzy membership values of network vertices. The simplicity of the proposed scheme enables efficient modifications, reducing computational complexity to a linear function of the network size and the number of communities. Moreover, to further reduce the complexity of FFMM for large size networks, the multi-cycle FFMM is proposed. Multi-cycle FFMM reduces complexity by breaking network into multiple sub-networks and applying FFMM to detect their communities. Performance of the proposed techniques are studied exploiting real-world data sets and the Lancichinetti-Fortunato-Radicchi (LFR) benchmark networks. Results proves that the multi-cycle FFMM produces a remarkable performance in terms of overlapping modularity, computational time, number of detected overlapping nodes, and *Overlapping Normalized Mutual Information* (ONMI).

Range-Free Anchor Selection via Community Detection In this study, a variety of community detection approaches include non-overlapping and overlapping methods were applied to the distributed localization problem in dense WSNs. As a novel application of community detection, a range-free anchor selection approach was proposed that can utilize both overlapping and also non-overlapping community detection. The proposed technique used a proposed community membership distance objective function to select the best subset of anchor nodes among a set of candidate

nodes in a network. Simulation results over a variety of networks illustrated that the proposed technique is efficient and effective, especially in networks with low relative range measurement noise where location ambiguity is the largest driver of localization error.

7.1 Future Work

The following ideas are candidates for future research on the topics addressed in this dissertation.

Completely Positive Programming Future work will focus on studying the interaction of the convex completely-positive problem and the rank-1 constraint, with an aim to develop a more efficient solution for finding max-modularity communities. Furthermore, we plan to improve computing efficiency of our algorithm for tackling large real-world social network data.

Hybrid Label Propagation In this study, the proposed HLP was applied to real-world data sets and benchmark networks to detect non-overlapping communities. Meanwhile, some of these networks included overlapping communities which cannot be detected with non-overlapping label propagation methods, such as the approaches

in this chapter. Therefore, in the future, we will develop HLP-based overlapping label propagation methods. This future work will have important implications for problems where discovery of overlapping community structure is important.

Linear-Time Modularity Gain Acceleration The results on real-world data sets validated the linear computational complexity of MGA. This opens a new era for all LP-based community detection techniques for very large data sets, where previous approaches were prone to fail due to very high computational complexity. Moreover, applications of the proposed criteria on overlapping LP-based approaches can be considered, which is left for future study.

Multi-Cycle FFMM Multi-cycle FFMM introduces vast potential for research incorporating applications of parallel processing for fuzzy community detection in large networks. Moreover, studying other distributions (rather than uniform) or methods to calculate the number of target communities of detected sub-networks is another open problem for future work.

Range-Free Anchor Selection via Community Detection In this chapter, application of community detection for range-free anchor selection with distributed localization was studied. However, cooperative (centralized) localization enables many more possible anchor/sensor combinations. Exploiting community detection for this

problem opens a new area of research in smart-ranging to select the best set of anchors/sensors for effective cooperative localization. Furthermore, community detection could be used to cluster a large network into multiple sub-networks for optimal hybrid cooperative localization. Also tuning weight coefficient γ according to machine learning approaches is desired.

References

[1] J. Su and T. C. Havens, "Fuzzy community detection in social networks using a genetic algortihm," in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, July 2014, pp. 2039–2046.

[2] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E*, vol. 76, no. 3, p. 036106, 2007.

[3] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[4] S. Yazdanparast and T. C. Havens, "Modularity maximization using completely positive programming," *Physica A: Statistical Mechanics and its Applications*, vol. 471, pp. 20–32, 2017.

[5] J. Piero, A. Berenstein, A. Gonzalez-Perez, A. Chernomoretz, and L. I. Furlong,

"Uncovering disease mechanisms through network biology in the era of next generation sequencing," *Scientific Reports 6,*, 2016.

[6] R. Ding, N. Ujang, H. b. Hamid, and J. Wu, *Complex Network Theory Applied to the Growth of Kuala Lumpurs Public Urban Rail Transit Network.* Public Library of Science, 10 2015, vol. 10.

[7] L. Weng, F. Menczer, and Y.-Y. Ahn, "Virality prediction and community structure in social networks," *Scientific Reports volume 3*, 2013.

[8] D. Dardari, A. Conti, C. Buratti, and R. Verdone, "Mathematical evaluation of environmental monitoring estimation error through energy-efficient wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 6, no. 7, pp. 790–802, 2007.

[9] E. Cayirci, H. Tezcan, Y. Dogan, and V. Coskun, "Wireless sensor networks for underwater survelliance systems," *Ad Hoc Networks*, vol. 4, no. 4, pp. 431 – 446, 2006.

[10] J. Wu, S. Yuan, S. Ji, G. Zhou, Y. Wang, and Z. Wang, "Multi-agent system design and evaluation for collaborative wireless sensor network in large structure health monitoring," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2028–2036, 2010.

[11] M. Jamalabdollahi and S. R. Zekavat, "OFDMA-based high resolution sensor node ToA estimation in non-homogenous medium of human body," in *2016 10th*

International Symposium on Medical Information and Communication Technology (ISMICT), March 2016, pp. 1–5.

[12] W. Xue, W. Sheng, and B. Daowei, "Distributed visual-target-surveillance system in wireless sensor networks," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 39, no. 5, pp. 1134–1146, 2009.

[13] J. Zhou, C. L. P. Chen, L. Chen, and W. Zhao, "A user-customizable urban traffic information collection method based on wireless sensor networks," *Intelligent Transportation Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–10, 2013.

[14] M. Jamalabdollahi and S. Zekavat, "ToA ranging and layer thickness computation in nonhomogeneous media," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 742–752, Feb 2017.

[15] S. M. George, Z. Wei, H. Chenji, W. Myounggyu, L. Yong Oh, A. Pazarloglou, R. Stoleru, and P. Barooah, "Distressnet: a wireless ad hoc and sensor network architecture for situation management in disaster response," *Communications Magazine, IEEE*, vol. 48, no. 3, pp. 128–136, 2010.

[16] L. Qing, T. Zhi, Y. Yuejun, and L. Yue, "Localized structural health monitoring using energy-efficient wireless sensor networks," *Sensors Journal, IEEE*, vol. 9, no. 11, pp. 1596–1604, 2009.

[17] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75 – 174, 2010.

[18] Z. Gao and N. Jin, "Detecting community structure in complex networks based on k-means clustering and data field theory," in *Control and Decision Conference, 2008. CCDC 2008. Chinese*. IEEE, Conference Proceedings, pp. 4411–4416.

[19] Y. van Gennip, B. Hunter, R. Ahn, P. Elliott, K. Luh, M. Halvorson, S. Reid, M. Valasik, J. Wo, and G. E. Tita, "Community detection using spectral clustering on sparse geosocial data," *SIAM Journal on Applied Mathematics*, vol. 73, no. 1, pp. 67–83, 2013.

[20] Y. Jiang, C. Jia, and J. Yu, "An efficient community detection method based on rank centrality," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 9, pp. 2182–2194, 2013.

[21] S. Jia, L. Gao, Y. Gao, and H. Wang, "Anti-triangle centrality-based community detection in complex networks," *IET systems biology*, vol. 8, no. 3, pp. 116–125, 2014.

[22] J. Cheng, L. Li, M. Leng, W. Lu, Y. Yao, and X. Chen, *A divisive spectral method for network community detection*, vol. 2016, no. 3, p. 033403.

[23] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical review E*, vol. 69, no. 6, p. 066133, 2004.

[24] G. Agarwal and D. Kempe, "Modularity-maximizing graph communities via mathematical programming," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 66, no. 3, pp. 409–418, 2008.

[25] V. d. F. Vieira, C. R. Xavier, N. F. Ebecken, and A. G. Evsukoff, *Modularity Based Hierarchical Community Detection in Networks*. Springer, 2014, pp. 146–160.

[26] T. Evans and R. Lambiotte, "Line graphs, link partitions, and overlapping communities," *Physical Review E*, vol. 80, no. 1, p. 016105, 2009.

[27] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Phys. Rev. E*, vol. 72, p. 027104, Aug 2005.

[28] D. Jin, D. He, D. Liu, and C. Baquero, "Genetic algorithm with local search for community mining in complex networks," in *Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on*, vol. 1. IEEE, Conference Proceedings, pp. 105–112.

[29] J. Liu, "Fuzzy modularity and fuzzy community structure in networks," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 77, no. 4, pp. 547–557, 2010.

[30] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," in *Advances in Neural Information Processing Systems*, Conference Proceedings, pp. 33–40.

[31] J. Su and T. C. Havens, *A Generalized Fuzzy T-norm Formulation of Fuzzy Modularity for Community Detection in Social Networks.* Springer, 2014, pp. 65–76.

[32] T. Cai and X. Li, "Robust and computationally feasible community detection in the presence of arbitrary outlier nodes," *arXiv preprint arXiv:1404.6000*, 2014.

[33] T. Nepusz, A. Petrczi, L. Ngyessy, and F. Bazs, "Fuzzy communities and the concept of bridgeness in complex networks," *Physical Review E*, vol. 77, no. 1, p. 016107, 2008.

[34] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical review E*, vol. 69, no. 2, p. 026113, 2004.

[35] L. Danon, A. Díaz-Guilera, and A. Arenas, *The effect of size heterogeneity on community identification in complex networks.* IOP Publishing, 2006, vol. 2006, no. 11, p. P11010.

[36] T. C. Havens, J. C. Bezdek, C. Leckie, K. Ramamohanarao, and M. Palaniswami, "A soft modularity function for detecting fuzzy communities in social networks," *Fuzzy Systems, IEEE Transactions on*, vol. 21, no. 6, pp. 1170–1175, 2013.

[37] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Physical review E*, vol. 72, no. 2, p. 027104, 2005.

[38] M. J. Barber and J. W. Clark, "Detecting network communities by propagating labels under constraints," *Physical Review E*, vol. 80, no. 2, p. 026129, 2009.

[39] T. Chao, N. Jianwei, W. Jinming, X. Zhongyu, and P. Fu, "Weighted label propagation algorithm for overlapping community detection," in *Communications (ICC), 2015 IEEE International Conference on*, 2015, Conference Proceedings, pp. 1238–1243.

[40] C. Staudt and H. Meyerhenke, "Engineering parallel algorithms for community detection in massive networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.

[41] S. Bandyopadhyay, G. Chowdhary, and D. Sengupta, "Focs: Fast overlapped community search," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 2974–2985, 2015.

[42] W. Cui, Y. Xiao, H. Wang, Y. Lu, and W. Wang, "Online search of overlapping communities," in *Proceedings of the 2013 ACM SIGMOD international conference on Management of data*. ACM, 2013, pp. 277–288.

[43] L. Speidel, T. Takaguchi, and N. Masuda, *Community detection in directed acyclic graphs*. Springer, 2015, vol. 88, no. 8, p. 203.

[44] Y. Sun, B. Danila, K. Josić, and K. E. Bassler, *Improved community structure detection using a modified fine-tuning strategy*. IOP Publishing, 2009, vol. 86, no. 2, p. 28004.

[45] W. Sun, Z. Wang, M. Jamalabdollahi, and S. A. R. Zekavat, "Experimental study on the difference between acoustic communication channels in freshwater rivers/lakes and in oceans," in *2014 48th Asilomar Conference on Signals, Systems and Computers*, Nov 2014, pp. 333–337.

[46] A. Pascale, M. Nicoli, F. Deflorio, B. Dalla Chiara, and U. Spagnolini, "Wireless sensor networks for traffic management and road safety," *Intelligent Transport Systems, IET*, vol. 6, no. 1, pp. 67–77, 2012.

[47] H. Will, K. Schleiser, and J. Schiller, "A real-time kernel for wireless sensor networks employed in rescue scenarios," in *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*, Conference Proceedings, pp. 834–841.

[48] Lo, x, G. pez, V. Custodio, and J. I. Moreno, "Lobin: E-textile and wireless-sensor-network-based platform for healthcare monitoring in future hospital environments," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 6, pp. 1446–1458, 2010.

[49] M. Jamalabdollahi, S. Zekavat, and K. Pahlavan, "High resolution OFDM-based sensor node ranging within in-homogeneous media of human body," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2019.

[50] I. F. Akyildiz, T. Melodia, and K. R. Chowdury, "Wireless multimedia sensor networks: A survey," *Wireless Communications, IEEE*, vol. 14, no. 6, pp. 32–39, 2007.

[51] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422.

[52] V. Kaseva, T. D. Hamalainen, and M. Hannikainen, "Range-free algorithm for energy-efficient indoor localization in wireless sensor networks," in *Design and Architectures for Signal and Image Processing (DASIP), 2011 Conference on*, Conference Proceedings, pp. 1–8.

[53] X. Qingjun, X. Bin, C. Jiannong, and W. Jianping, "Multihop range-free localization in anisotropic wireless sensor networks: A pattern-driven scheme," *Mobile Computing, IEEE Transactions on*, vol. 9, no. 11, pp. 1592–1607, 2010.

[54] M. Jamalabdollahi and S. Zekavat, "Energy efficient ranging in wireless sensor networks via a new time slot-based round-trip algorithm," in *Aerospace Conference, 2014 IEEE*, Conference Proceedings, pp. 1–7.

[55] Z. Sahinoglu and S. Gezici, "Ranging in the ieee 802.15.4a standard," in *Wireless and Microwave Technology Conference, 2006. WAMICON '06. IEEE Annual*, Conference Proceedings, pp. 1–5.

[56] C. Alippi and G. Vanini, "A rssi-based and calibrated centralized localization technique for wireless sensor networks," in *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*, Conference Proceedings, pp. 5 pp.–305.

[57] M. Jamalabdollahi and S. A. R. Zekavat, "Joint neighbor discovery and time of arrival estimation in wireless sensor networks via OFDMA," *Sensors Journal, IEEE*, vol. 15, no. 10, pp. 5821–5833, 2015.

[58] A. Tsalach, I. Steinberg, and I. Gannot, "Time difference of arrival based cancer tumor localization using magnetic nano-particles induced acoustic signals," Thesis, 2012.

[59] P. Rong and M. L. Sichitiu, "Angle of arrival localization for wireless sensor networks," in *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, vol. 1, Conference Proceedings, pp. 374–382.

[60] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, "Mac essentials for wireless sensor networks," *Communications Surveys and Tutorials, IEEE*, vol. 12, no. 2, pp. 222–248, 2010.

[61] S. Hong, D. Zhi, S. Dasgupta, and Z. Chunming, "Multiple source localization in wireless sensor networks based on time of arrival measurement," *Signal Processing, IEEE Transactions on*, vol. 62, no. 8, pp. 1938–1949, 2014.

[62] Y. Kehu, W. Gang, and L. Zhi-Quan, "Efficient convex relaxation methods for robust target localization by a sensor network using time differences of arrivals," *Signal Processing, IEEE Transactions on*, vol. 57, no. 7, pp. 2775–2784, 2009.

[63] M. Jamalabdollahi and S. A. R. Zekavat, "Joint neighbor discovery and time of arrival estimation in wireless sensor networks via OFDMA," *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5821–5833, Oct 2015.

[64] K. Steinhaeuser and N. V. Chawla, "Identifying and evaluating community structure in complex networks," *Pattern Recognition Letters*, pp. 413–421, 2010.

[65] M. Ovelgnne, A. Geyer-Schulz, and M. Stein, "Randomized greedy modularity optimization for group detection in huge social networks," in *Proceedings of the fourth SNA-KDD Workshop, KDD 2010, July*, vol. 25, Conference Proceedings, pp. 1–9.

[66] Y. Zhang and D.-Y. Yeung, "Overlapping community detection via bounded nonnegative matrix tri-factorization," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, Conference Proceedings, pp. 606–614.

[67] U. Brandes, D. Delling, M. Gaertler, R. Grke, M. Hoefer, Z. Nikoloski, and D. Wagner, "Maximizing modularity is hard," *arXiv preprint physics/0608255*, 2006.

[68] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[69] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 20, no. 2, pp. 172–188, 2008.

[70] S. Gregory, "Fuzzy overlapping communities in networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2011, no. 02, p. P02017, 2011.

[71] S. Burer, "Optimizing a polyhedral-semidefinite relaxation of completely positive programs," *Mathematical Programming Computation*, vol. 2, no. 1, pp. 1–19.

[72] J. Keller, R. Krisnapuram, and N. R. Pal, *Fuzzy models and algorithms for pattern recognition and image processing*. Springer Science and Business Media, 2005, vol. 4.

[73] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[74] S. Burer, *Copositive programming*. Springer, 2012, pp. 201–218.

[75] J. F. Sturm, "Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones," *Optimization methods and software*, vol. 11, no. 1-4, pp. 625–653, 1999.

[76] Z. Wen, D. Goldfarb, and W. Yin, "Alternating direction augmented lagrangian methods for semidefinite programming," *Mathematical Programming Computation*, vol. 2, no. 3-4, pp. 203–230, 2010.

[77] M. V. Ramana, L. Tunel, and H. Wolkowicz, "Strong duality for semidefinite programming," *SIAM Journal on Optimization*, vol. 7, no. 3, pp. 641–662, 1997.

[78] L. Yang, D. Sun, and K.-C. Toh, "Sdpnal +: A majorized semismooth newton-cg augmented lagrangian method for semidefinite programming with nonnegative constraints," *arXiv preprint arXiv:1406.0942*, 2014.

[79] J.-J. Moreau, "Dcomposition orthogonale dun espace hilbertien selon deux cnes mutuellement polaires," *CR Acad. Sci. Paris*, vol. 255, pp. 238–240, 1962.

[80] W. Li, "Revealing network communities with a nonlinear programming method," *Information Sciences*, vol. 229, pp. 18–28, 2013.

[81] B. Yang, J. Liu, and J. Feng, "On the spectral characterization and scalable mining of network communities," *Knowledge and Data Engineering, IEEE Transactions On*, vol. 24, no. 2, pp. 326–337, 2012.

[82] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, vol. 23, no. 2, pp. 298–305, 1973.

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

[84] J. Shi and J. Malik, "Normalized cuts and image segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 888–905, 2000.

[85] R. Guimera and L. A. N. Amaral, "Cartography of complex networks: modules and universal roles," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, no. 02, p. P02001, 2005.

[86] J. C. Bezdek and R. J. Hathaway, "Vat: A tool for visual assessment of (cluster) tendency," in *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, vol. 3. IEEE, Conference Proceedings, pp. 2225–2230.

[87] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, pp. 452–473, 1977.

[88] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behavioral Ecology and Sociobiology*, vol. 54, no. 4, pp. 396–405, 2003.

[89] D. E. Knuth, D. E. Knuth, and D. E. Knuth, *The Stanford GraphBase: a platform for combinatorial computing*. Addison-Wesley Reading, 1993, vol. 37.

[90] P. M. Gleiser and L. Danon, "Community structure in jazz," *Advances in complex systems*, vol. 6, no. 04, pp. 565–573, 2003.

[91] M. Gong, J. Liu, L. Ma, Q. Cai, and L. Jiao, "Novel heuristic density-based method for community detection in networks," *Physica A: Statistical Mechanics and its Applications*, vol. 403, pp. 71 – 84, 2014.

[92] R. Guimera and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, no. 7028, pp. 895–900, 2005.

[93] G. Golub and C. Van Loan, "The pseudo-inverse," *Matrix Computations (3rd edition), The Johns Hopkins University Press*, pp. 257–258, 1996.

[94] C. Staudt and H. Meyerhenke, "Engineering parallel algorithms for community detection in massive networks."

[95] A. Lancichinetti and S. Fortunato, "Community detection algorithms: a comparative analysis," *Physical review E*, vol. 80, no. 5, p. 056117, 2009.

[96] G. Tibly and J. Kertsz, "On the equivalence of the label propagation method of community detection and a potts model approach," *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 1920, pp. 4982–4984.

[97] G. Cordasco and L. Gargano, "Community detection via semi-synchronous label propagation algorithms," in *Business Applications of Social Network Analysis (BASNA), 2010 IEEE International Workshop on*, Conference Proceedings, pp. 1–8.

[98] L. Xin and T. Murata, "Community detection in large-scale bipartite networks," in *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT '09. IEEE/WIC/ACM International Joint Conferences on*, vol. 1, Conference Proceedings, pp. 50–57.

[99] M. He, M. Leng, F. Li, Y. Yao, and X. Chen, *A Node Importance Based Label Propagation Approach for Community Detection*. Springer, 2014, pp. 249–257.

[100] X. Jierui and B. K. Szymanski, "Community detection using a neighborhood strength driven label propagation algorithm," in *Network Science Workshop (NSW), 2011 IEEE*, Conference Proceedings, pp. 188–195.

[101] X. Liu and T. Murata, "Advanced modularity-specialized label propagation algorithm for detecting communities in networks," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 7, pp. 1493–1500, 2010.

[102] K. Berahmand and A. Bouyer, *LP-LPA: A link influence-based label propagation algorithm for discovering community structures in networks*, no. 06, p. 1850062, exported from https://app.dimensions.ai on 2018/10/25.

[103] M. Leng, Y. Yao, J. Cheng, W. Lv, and X. Chen, "Active semi-supervised community detection algorithm with label propagation," in *Database Systems for Advanced Applications*. Springer, Conference Proceedings, pp. 324–338.

[104] X. Jierui and B. K. Szymanski, "Labelrank: A stabilized label propagation

algorithm for community detection in networks," in *Network Science Workshop (NSW), 2013 IEEE 2nd*, Conference Proceedings, pp. 138–143.

[105] D. Qiguo, G. Maozu, L. Yang, L. Xiaoyan, and C. Ling, "Mlpa: Detecting overlapping communities by multi-label propagation approach," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, Conference Proceedings, pp. 681–688.

[106] S. Song, C. Yuzhong, F. Mingyue, L. Wanhua, and Shining, "A hierarchical multi-label propagation algorithm for overlapping community discovery in social networks," in *Web Information System and Application Conference (WISA), 2014 11th*, Conference Proceedings, pp. 113–118.

[107] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti, "Generalized louvain method for community detection in large networks," in *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*, 2011, Conference Proceedings, pp. 88–93.

[108] M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some simplified np-complete problems," in *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*, ser. STOC '74, 1974, pp. 47–63.

[109] J. Hou Chin and K. Ratnavelu, "A semi-synchronous label propagation algorithm with constraints for community detection in complex networks," *Scientific Reports*, vol. 7, no. 45836.

[110] H. Meyerhenke, P. Sanders, and C. Schulz, "Parallel graph partitioning for complex networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 9, pp. 2625–2638, Sept 2017.

[111] S. Qiao, N. Han, Y. Gao, R. Li, J. Huang, J. Guo, L. A. Gutierrez, and X. Wu, "A fast parallel community discovery model on complex networks through approximate optimization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1638–1651, Sept 2018.

[112] K. Berahmand, A. Bouyer, and M. Vasighi, "Community detection in complex networks by detecting and expanding core nodes through extended local similarity of nodes," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 4, pp. 1021–1033, Dec 2018.

[113] Y. Xing, F. Meng, Y. Zhou, M. Zhu, M. Shi, and G. Sun, "A node influence based label propagation algorithm for community detection in networks," *The Scientific World Journal*, vol. 2014, 2014.

[114] M. Rosvall and C. Bergstrom, *Maps of information flow reveal community structure in complex networks*, 2007.

[115] A. Lancichinetti, S. Fortunato, and F. Radicchi, *Benchmark graphs for testing community detection algorithms.* American Physical Society, Oct 2008, vol. 78, p. 046110.

[116] A. Lancichinetti and S. Fortunato, *Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities.* American Physical Society, Jul 2009, vol. 80, p. 016118.

[117] L. Danon, A. Daz-Guilera, J. Duch, and A. Arenas, *Comparing community structure identification*, vol. 2005, no. 09, p. P09008.

[118] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, p. 78217826, 2002.

[119] A. D.-G. F. G. R. Guimer'a, L. Danon and A. Arenas, "Self-similar community structure in a network of human interactions," *Phys. Rev. E*, vol. 68, 2003.

[120] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 539–547.

[121] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," %urlhttp://snap.stanford.edu/data, Jun. 2014.

[122] S. Emmons, S. Kobourov, M. Gallant, and K. Brner, *Analysis of Network Clustering Algorithms and Cluster Quality Metrics at Scale.* Public Library of Science, 07 2016, vol. 11.

[123] C. Pizzuti, "A multiobjective genetic algorithm to find communities in complex networks," *IEEE Transactions on Evolutionary Computation*, vol. 16, pp. 418–430, 2012.

[124] P. Bedi and C. Sharma, "Community detection in social networks," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 6*, vol. 3, pp. 115–135, 2016.

[125] J. Su and T. C. Havens, "Quadratic program-based modularity maximization for fuzzy community detection in social networks," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 5, pp. 1356–1371, 2015.

[126] W. Luo, Z. Yan, C. Bu, and D. Zhang, "Community detection by fuzzy relations," *IEEE Transactions on Emerging Topics in Computing*, 2017.

[127] F. Breve and L. Zhao, "Fuzzy community structure detection by particle competition and cooperation," *Soft Computing*, vol. 17, no. 4, pp. 659–673, 2013.

[128] N. Binesh and M. Rezghi, "Fuzzy clustering in community detection based on nonnegative matrix factorization with two novel evaluation criteria," *Applied Soft Computing*, 2017.

[129] H. Zhang, X. Chen, J. Li, and B. Zhou, "Fuzzy community detection via modularity guided membership-degree propagation," *Pattern Recognition Letters*, vol. 70, pp. 66–72, 2016.

[130] P. Chopade and J. Zhan, "A framework for community detection in large networks using game-theoretic modeling," *IEEE Transactions on Big Data*, vol. 3, no. 3, pp. 276–288, Sep. 2017.

[131] H. K. Shakya, K. Singh, and B. Biswas, "An efficient genetic algorithm for fuzzy community detection in social network," in *Advanced Informatics for Computing Research*. Springer, 2017, Book Section, pp. 63–72.

[132] M. Hajiabadi, H. Zare, and H. Bobarshad., "An integrated approach for overlapping and non-overlapping community detection," *Knowledge-Based Systems*, vol. 123, pp. 188 – 199, 2017.

[133] X. L. W. Wang, D. Liu and L. Pan, "Fuzzy overlapping community detection based on local random walk and multidimensional scaling," *Physica A*, vol. 392, p. 6578 6586, 2013.

[134] A. Biswas and B. Biswas, "Fuzag: Fuzzy agglomerative community detection by exploring the notion of self-membership," *IEEE Transactions on Fuzzy Systems*, vol. PP, no. 99, pp. 1–1.

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

[136] A. F. McDaid, D. Greene, and N. Hurley, "Normalized mutual information to evaluate overlapping community finding algorithms," *arXiv preprint arXiv:1110.2515*, 2011.

[137] M. Jamalabdollahi and S. R. Zekavat, "High resolution ToA estimation via optimal waveform design," *IEEE Transactions on Communications*, vol. 65, no. 3, pp. 1207–1218, March 2017.

[138] S. Lee, B. Koo, and S. Kim, "Raps: reliable anchor pair selection for range-free localization in anisotropic networks," *IEEE Communications Letters*, vol. 18, no. 8, pp. 1403–1406, 2014.

[139] H. Woo and C. Lee, "A novel multihop range-free localization algorithm based on reliable anchor selection in wireless sensor networks." *KSII Transactions on Internet & Information Systems*, vol. 10, no. 2, 2016.

[140] M. Šimek, "Reference nodes selection for anchor-free localization in wireless sensor networks," *Vysoké učení technické v Brně, ČR*, 2010.

[141] O. Oshiga, X. Chu, Y.-W. Leung, and J. Ng, "Anchor selection for localization in large indoor venues," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. IEEE, 2018, pp. 1–6.

[142] P. Zhang and Q. Wang, "Anchor selection with anchor location uncertainty in wireless sensor network localization," in *2011 IEEE International Conference*

on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2011, pp. 4172–4175.

[143] Q. Chang, H. T. Hou, X. H. Zeng, Q. Li, and W. P. Wang, "An anchor selection algorithm in wireless sensor network," in *Applied Mechanics and Materials*, vol. 380. Trans Tech Publ, 2013, pp. 3962–3965.

[144] H. Woo and C. Lee, "Geometric range-free localization algorithm based on optimal anchor node selection in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 10, no. 4, p. 509892, 2014.

[145] I. Guvenc, S. Gezici, F. Watanabe, and H. Inamura, "Enhancements to linear least squares localization through reference selection and ml estimation," in *2008 IEEE Wireless Communications and Networking Conference*. IEEE, 2008, pp. 284–289.

[146] Y. Wang, F. Zheng, M. Wiemeler, W. Xiong, and T. Kaiser, "Reference selection for hybrid toa/rss linear least squares localization," in *2013 IEEE 78th vehicular technology conference (VTC Fall)*. IEEE, 2013, pp. 1–5.

[147] S. Wu, J. Li, and S. Liu, "Improved localization algorithms based on reference selection of linear least squares in los and nlos environments," *Wireless personal communications*, vol. 68, no. 1, pp. 187–200, 2013.

[148] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding

of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[149] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110.

Appendix A

Proof of Proposed Propositions at Chapter 2

A.0.1 Proposition 1

The adjoint operator of linear operator $\mathcal{A}(\mathbf{X}) := diag(\mathbf{A}\mathbf{X}\mathbf{A}^T) = \mathbf{b}^2$ is $\mathcal{A}^*(y) := \mathbf{A}^T diag(\mathbf{y})\mathbf{A}$.

$$
\begin{aligned}
\langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) \rangle &= \sum_{i,j=1}^{c\times n} \mathbf{X}_{ij} \left(\mathcal{A}^*(\mathbf{y})\right)_{ji}, \\
&= \sum_{i,j=1}^{c\times n} \mathbf{X}_{ij} \sum_{k=1}^{n} \mathbf{A}_{ki} y_k \mathbf{A}_{kj}, \\
&= \sum_{k=1}^{n} \mathbf{y}_k \sum_{i,j=1}^{c\times n} \mathbf{A}_{ki} \mathbf{X}_{ij} A_{kj} = \langle \mathcal{A}(\mathbf{X}), \mathbf{y} \rangle .
\end{aligned} \tag{A.1}
$$

A.0.2 Proposition 2

The solution for $\mathbf{y}$ in the dual Lagrangian at (2.18) is

$$
\mathbf{y} = \left(\mathcal{A}\mathcal{A}^*\right)^{-1} \left[\left(\mathbf{b}^T - \mathcal{A}\left(\mathbf{X}\right)\right)\mu - \mathcal{A}\left(\mathbf{S} - \mathbf{C}\right)\right]. \tag{A.2}
$$

We begin by differentiating (2.18) with respect to $\mathbf{y}$:

$$
\begin{aligned}
\nabla_y L(\mathbf{y}, \mathbf{S}, \mathbf{X}) = -\mathbf{b}^T &+ \frac{\partial}{\partial \mathbf{y}} \langle \mathbf{X}, \mathcal{A}^*(\mathbf{y}) + \mathbf{S} - \mathbf{C} \rangle , \\
&+ \frac{\partial}{\partial y} 1/(2\mu) \left\| \mathcal{A}^*(\mathbf{y}) + \mathbf{S} - \mathbf{C} \right\|^2 ,
\end{aligned} \tag{A.3}
$$

where

$$\begin{aligned}
&\frac{\partial}{\partial y}\left\langle \mathbf{X}, \mathcal{A}^{*}(\mathbf{y})+\mathbf{S}-\mathbf{C}\right\rangle = \\
&\frac{\partial}{\partial y}\left\{\sum_{i,j=1}^{c\times n} X_{ij}\left(\mathcal{A}^{*}(y)\right)_{ji}+\sum_{i,j=1}^{c\times n}\mathbf{X}_{ij}\left(\mathbf{S}-\mathbf{C}\right)_{ji}\right\}, \\
&=\frac{\partial}{\partial y}\left\{\sum_{i,j=1}^{c\times n}\mathbf{X}_{ij}\sum_{k=1}^{n}\mathbf{A}_{ki}\mathbf{y}_{k}\mathbf{A}_{kj}\right\}, \\
&=\sum_{i,j=1}^{c\times n}\mathbf{A}_{ki}X_{ij}\mathbf{A}_{kj}=\mathcal{A}\left(\mathbf{X}\right).
\end{aligned} \tag{A.4}$$

and

$$\begin{aligned}
&\frac{\partial}{\partial y}\left\{1/(2\mu)\left\|\mathcal{A}^{*}(\mathbf{y})+\mathbf{S}-\mathbf{C}\right\|^{2}\right\} = \\
&\frac{1}{2\mu}\frac{\partial}{\partial \mathbf{y}}\sum_{i,j=1}^{c\times n}\left(\sum_{k=1}^{n}\mathbf{A}_{ki}\mathbf{y}_{k}\mathbf{A}_{kj}+\mathbf{S}_{ij}-\mathbf{C}_{ij}\right)^{2}, \\
&=\frac{1}{2\mu}\frac{\partial}{\partial \mathbf{y}}\sum_{i,j=1}^{c\times n}\left[\left(\sum_{k=1}^{n}\mathbf{A}_{ki}\mathbf{y}_{k}\mathbf{A}_{kj}\right)^{2}+\right. \\
&\left.2\sum_{k=1}^{n}\mathbf{A}_{ki}\mathbf{y}_{k}\mathbf{A}_{kj}\left(\mathbf{S}_{ij}-\mathbf{C}_{ij}\right)+\left(\mathbf{S}_{ij}-\mathbf{C}_{ij}\right)^{2}\right],
\end{aligned} \tag{A.5}$$

$$\begin{aligned}
&=\frac{1}{2\mu}\sum_{i,j=1}^{c\times n}\left[2\left(\sum_{k=1}^{n}\mathbf{A}_{ki}\mathbf{y}_{k}\mathbf{A}_{kj}\right)\mathbf{A}_{ki}\mathbf{A}_{kj}+2\mathbf{A}_{ki}\mathbf{A}_{kj}\left(\mathbf{S}_{ij}-\mathbf{C}_{ij}\right)\right], \\
&=\frac{1}{\mu}\left[\mathcal{A}\mathcal{A}^{*}\left(\mathbf{y}\right)+\mathcal{A}\left(\mathbf{S}-\mathbf{C}\right)\right].
\end{aligned} \tag{A.6}$$

Combining (A.4) and (A.5) results in

$$\nabla_{\mathbf{y}}L(\mathbf{y},\mathbf{S},\mathbf{X})=-\mathbf{b}^{T}+\mathcal{A}\left(\mathbf{X}\right)+\frac{1}{\mu}\left[\mathcal{A}\mathcal{A}^{*}\left(\mathbf{y}\right)+\mathcal{A}\left(\mathbf{S}-\mathbf{C}\right)\right]=0. \tag{A.7}$$

Solving for $\mathbf{y}$ completes the proof.

A.0.3 Proposition 3

The optimum solution for $\mathbf{S}$ in (2.18) is obtained by the optimization

$$\mathbf{S}^* = \arg\min_{\mathbf{S}} \left\{ \|\mathbf{S} - \mathbf{V}(\mathbf{y}, \mathbf{X})\|_F^2 \right\}, \ \mathbf{S} \succeq 0, \tag{A.8}$$

where $\mathbf{V}(\mathbf{y}, \mathbf{X}) = (\mathbf{C} - \mathcal{A}^*(\mathbf{y}) - \mu\mathbf{X})$.

Minimizing (2.18) with respect to $\mathbf{S}$ gives

$$\begin{aligned}
&\min_{\mathbf{S}} \{L(\mathbf{S}|\mathbf{y}, \mathbf{X})\} = \\
&\min_{\mathbf{S}} \left\{ \sum_{i,j=1}^{c\times n} \mathbf{X}_{ij}\mathbf{S}_{ij} + \frac{1}{2\mu} \sum_{i,j=1}^{c\times n} \left((\mathcal{A}^*(\mathbf{y}))_{ij} + \mathbf{S}_{ij} - \mathbf{C}_{ij} \right)^2 \right\}, \\
&= \min_{\mathbf{S}} \left\{ \frac{1}{2\mu} \sum_{i,j=1}^{c\times n} \left[2\left(\mu\mathbf{X}_{ij} + (\mathcal{A}^*(\mathbf{y}))_{ij} - \mathbf{C}_{ij} \right) \mathbf{S}_{ij} + \mathbf{S}_{ij}^2 \right] \right\}, \\
&= \min_{\mathbf{S}} \left\{ \sum_{i,j=1}^{c\times n} \left[2\left(\mu X_{ij} + (\mathcal{A}^*(\mathbf{y}))_{ij} - \mathbf{C}_{ij} \right) \mathbf{S}_{ij} + \mathbf{S}_{ij}^2 \right] \right\}.
\end{aligned} \tag{A.9}$$

Considering $\|\mathbf{S} - (\mathbf{C} - \mathcal{A}^*(\mathbf{y}) - \mu\mathbf{X})\|_F^2$, it is easy to show that

$$\begin{aligned}
&\min_{\mathbf{S}} \left\{ \|\mathbf{S} - (\mathbf{C} - \mathcal{A}^*(\mathbf{y}) - \mu\mathbf{X})\|_F^2 \right\} =, \\
&\min_{\mathbf{S}} \left\{ \sum_{i,j=1}^{c\times n} \left(\mathbf{S}_{ij} - (\mathbf{C} - \mathcal{A}^*(\mathbf{y}) - \mu\mathbf{X})_{ij} \right)^2 \right\}, \\
&= \min_{\mathbf{S}} \left\{ \sum_{i,j=1}^{c\times n} \left(2\mathbf{S}_{ij} \left(-\mathbf{C} + \mathcal{A}^*(\mathbf{y}) + \mu\mathbf{X} \right)_{ij} + \mathbf{S}_{ij}^2 \right) \right\}, \\
&= \min_{\mathbf{S}} \left\{ \sum_{i,j=1}^{c\times n} \left[2 \left(\mu\mathbf{X}_{ij} + (\mathcal{A}^*(\mathbf{y}))_{ij} - \mathbf{C}_{ij} \right) \mathbf{S}_{ij} + \mathbf{S}_{ij}^2 \right] \right\}.
\end{aligned} \tag{A.10}$$

which is the same as that derived in (A.9).

Appendix B

Proof of the Proposed Modularity Gain Objective Function at Chapter 4

In this section, we outline the proofs of the proposed modularity gain objective function at (4.6). Considering (4.2), the modularity gain (ΔQ) associated with changing the community membership (label) of the ith node from current label ℓ_i to the new label ℓ_j is

$$\Delta Q\left(i, \ell_i \to \ell_j\right) = \frac{\operatorname{tr}\left(\mathbf{U}_{\ell_i \to \ell_j}\mathbf{B}\mathbf{U}_{\ell_i \to \ell_j}^T - \mathbf{U}_{\ell_i}\mathbf{B}\mathbf{U}_{\ell_i}^T\right)}{\|\mathbf{W}\|}, \tag{B.1}$$

here $\mathbf{B}$ is defined at (4.2), and $\mathbf{U}_{\ell_i \to \ell_j}$ represents the partition matrix $\mathbf{U}$ defined at (4.2), when the community membership (label) ℓ_j is dedicated to the ith node. Also, $\mathbf{U}_{\ell_i}$ entails that label of ith node is ℓ_i. By substituting(4.4) into (4.5), we can write

$$\ell_i^{(k+1)} = \arg\max_{\ell_j} \Bigg\{ \mathrm{tr}(\mathbf{U}_{\ell_i \to \ell_j} \mathbf{B} {\mathbf{U}_{\ell_\mathbf{i} \to \ell_\mathbf{j}}}^T - \mathbf{U}_{\ell_i} \mathbf{B} \mathbf{U}_{\ell_i}^T) \Bigg\},\ \ell_j \in \mathcal{L}_i', \tag{B.2}$$

Here $\mathcal{L}_i'$ is the set of available membership labels, where $\mathcal{L}_i' = \{\ell_j,\ \forall j \in \mathcal{N}_i\}$. Specifying $\mathbf{U}_{\ell_i \to \ell_j} = \tilde{\mathbf{U}}_i + \check{\mathbf{U}}_{\ell_i \to \ell_j}$,

$$\tilde{\mathbf{U}}_i = [\mathbf{u}_1, ..., \mathbf{u}_{i-1}, \mathbf{0}_{c\times 1}, \mathbf{u}_{i+1}, ..., \mathbf{u}_N], \tag{B.3a}$$

$$\check{\mathbf{U}}_{\ell_i \to \ell_j} = [\mathbf{0}_{c\times 1}, ..., \mathbf{0}_{c\times 1}, \mathbf{e}, \mathbf{0}_{c\times 1}, ..., \mathbf{0}_{c\times 1}], \tag{B.3b}$$

where $\mathbf{e}$ is a $\mathbf{c} \times \mathbf{1}$ vector, where $\mathbf{e}_i = 1$, for $i = \ell_j$, and $\mathbf{e}_i = 0$ for $i \neq \ell_j$. Hence,

$$\begin{aligned} \arg\max_{\ell_j} \Bigg\{ \mathrm{tr}(\mathbf{U}_{\ell_i \to \ell_j} \mathbf{B} {\mathbf{U}_{\ell_\mathbf{i} \to \ell_\mathbf{j}}}^T - \mathbf{U}_{\ell_i} \mathbf{B} \mathbf{U}_{\ell_i}^T) \Bigg\} = \\ \arg\max_{\ell_j} \Bigg\{ \mathrm{tr} \Big(\tilde{\mathbf{U}}_i \mathbf{B} \tilde{\mathbf{U}}_i^T + \tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T + \check{\mathbf{U}}_{\ell_i \to \ell_j} \mathbf{B} \tilde{\mathbf{U}}_i^T + \\ \check{\mathbf{U}}_{\ell_i \to \ell_j} \mathbf{B} \check{\mathbf{U}}_{\ell_i \to \ell_j}^T - \tilde{\mathbf{U}}_i \mathbf{B} \tilde{\mathbf{U}}_i^T - \\ \tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}_{\ell_i}^T - \check{\mathbf{U}}_{\ell_i} \mathbf{B} \tilde{\mathbf{U}}_i^T - \check{\mathbf{U}}_{\ell_i} \mathbf{B} \check{\mathbf{U}}_{\ell_i}^T \Big) \Bigg\},\ \ell_j \in \mathcal{L}_i'. \end{aligned} \tag{B.4}$$

where $\mathbf{B}$, and $\mathcal{L}'_i$ are defined at (4.2) and (4.9), respectively, and $\tilde{\mathbf{U}}_i$ and $\check{\mathbf{U}}_{\ell_i \to \ell_j}$ are defined at (B.3). Applying simple mathematical manipulations it can be show that $\text{tr}\left(\check{\mathbf{U}}_{\ell_i \to \ell_j}\mathbf{B}\check{\mathbf{U}}^T_{\ell_i \to \ell_j}\right) = \text{tr}\left(\check{\mathbf{U}}_{\ell_i}\mathbf{B}\check{\mathbf{U}}^T_{\ell_i}\right) = b_{ii}$. Considering the very fact that b_{ii} is constant, then (B.4) is simplified to

$$\begin{aligned}\arg\max_{\ell_j}\{\text{tr}\Bigg(&\tilde{\mathbf{U}}_i\mathbf{B}\check{\mathbf{U}}^T_{\ell_i \to \ell_j} + \check{\mathbf{U}}_{\ell_i \to \ell_j}\mathbf{B}\tilde{\mathbf{U}}^T_i + \\ &- \tilde{\mathbf{U}}_i\mathbf{B}\check{\mathbf{U}}^T_{\ell_i} - \check{\mathbf{U}}_{\ell_i}\mathbf{B}\tilde{\mathbf{U}}^T_i\Bigg)\Bigg\},\ \ell_j \in \mathcal{L}'_i. \end{aligned} \tag{B.5}$$

In (B.5) $\tilde{\mathbf{U}}_i\mathbf{B}\check{\mathbf{U}}^T_{\ell_i \to \ell_j}$ and $\check{\mathbf{U}}_{\ell_i \to \ell_j}\mathbf{B}\tilde{\mathbf{U}}^T_i$ are the only components subject to change with respect to ℓ_j. Therefore, applying $\text{tr}(\mathbf{A}) = \text{tr}(\mathbf{A}^T)$ and the very fact that $\mathbf{B} = \mathbf{B}^T$, (B.5) is simplified to

$$\ell_i^{(k+1)} = \arg\max_{\ell_j}\left\{\text{tr}\left(\tilde{\mathbf{U}}_i\mathbf{B}\check{\mathbf{U}}^T_{\ell_i \to \ell_j}\right)\right\}, \ell_j \in \mathcal{L}'_i, \tag{B.6}$$

Since (B.6) maximizes label transitions from ℓ_i into ℓ_j for $\ell_j \in \mathcal{L}'_i$, this can lead to lower overall modularity. Therefore, in order to prevent transitions with negative impact, the proposed objective function at (B.7) is also evaluated for the current label ℓ_i, or

$$\ell_i^{(k+1)} = \arg\max_{\ell_j}\left\{\text{tr}\left(\tilde{\mathbf{U}}_i\mathbf{B}\check{\mathbf{U}}^T_{\ell_i \to \ell_j}\right)\right\}, \ell_j \in \mathcal{L}_i, \tag{B.7}$$

where $\mathcal{L}_i = \{\ell_i, \ell_j,\ \forall j \in \mathcal{N}_i\}$.

Substituting $b_{ni} = w_{ni} - \frac{m_n m_i}{\|\mathbf{W}\|}$, represented by the definition of modularity matrix proposed at (4.2), into $\text{tr}\left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}^T_{\ell_i \to \ell_j}\right) = \sum_{n \setminus \{i\}} u_{\ell_j n} b_{ni}$, leads to

$$\text{tr}\left(\tilde{\mathbf{U}}_i \mathbf{B} \check{\mathbf{U}}^T_{\ell_i \to \ell_j}\right) = \sum_{n \setminus \{i\}} u_{\ell_j n} w_{ni} - \frac{m_i}{\|\mathbf{W}\|} \sum_{n \setminus \{i\}} u_{\ell_j n} m_n. \tag{B.8}$$

Considering $w_{ni} = 0$ for $n \notin \mathcal{N}_i$, the modularity gain objective function can be evaluated by

$$\begin{aligned} \ell_i^{(k+1)} = \arg\max_{\ell_j} \Bigg\{ & \sum_{n \in \mathcal{N}_i} u_{\ell_j n} w_{ni} -, \\ & \frac{m_i}{\|\mathbf{W}\|} \sum_{n \setminus \{i\}} u_{\ell_j n} m_n \Bigg\}, \ell_j \in \mathcal{L}_i. \end{aligned} \tag{B.9}$$

Appendix C

Letter of Permission

Permission letter for content in Chapter 2.