

# COMMUNITY DETECTION IN COMPLEX NETWORK: METRIC SPACE, NEAREST NEIGHBOR SEARCH, LOW-RANK APPROXIMATION AND OPTIMALITY

*Thesis submitted in fulfilment of the requirements for the degree of*

## DOCTOR OF PHILOSOPHY

by

**Suman Saha**

under the supervision of

**Prof. S. P. Ghre**



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY  
Waknaghat, Solan, H.P.

· May 2016 ·

# Declarations

I hereby declare that the work reported in the Ph.D. thesis entitled “**COMMUNITY DETECTION IN COMPLEX NETWORK: METRIC SPACE, NEAREST NEIGHBOR SEARCH, LOW-RANK APPROXIMATION AND OPTIMALITY**” submitted at **Jaypee University of Information Technology, Wagnaghat, Solan, H.P., India** is an authentic record of my work carried out under the supervision of **Prof. S.P. Ghrera**. I have not submitted this work elsewhere for any other degree or diploma. I am fully responsible for the contents of my Ph.D Theses.



Suman Saha

Department of Computer Science and Engineering.

Jaypee University of Information Technology, Wagnaghat, Solan, H.P., India.

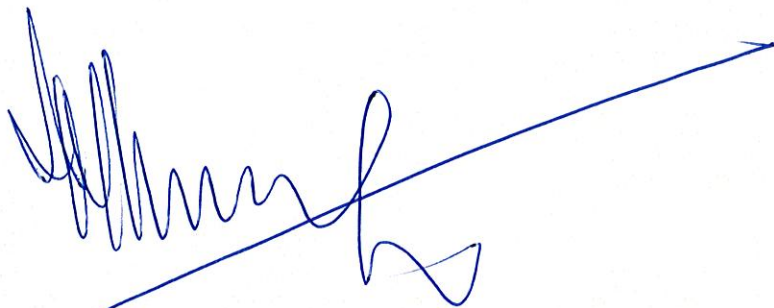
Date

**Copyright ©; 2016 by Suman Saha.**

“The copyright of this thesis rests with the author. No quotations from it should be published without the author’s prior written consent and information derived from it should be acknowledged.”

## Supervisor's Certificate

This is to certify that the work reported in the Ph.D. thesis entitled "COMMUNITY DETECTION IN COMPLEX NETWORK: METRIC SPACE, NEAREST NEIGHBOR SEARCH, LOW-RANK APPROXIMATION AND OPTIMALITY", submitted by Suman Saha at Jaypee University of Information Technology, Wagnaghat, Solan, H.P., India, is a bonafide record of his original work carried out under my supervision. This work has not been submitted elsewhere for any other degree or diploma.



Prof. S.P. Ghreera

Department of Computer Science and Engineering.

Jaypee University of Information Technology, Wagnaghat, Solan, H.P., India.

Date 11th Feb 2017



# Acknowledgments

Though only my name appears on the cover of this dissertation, a great many people have contributed to its production. I owe my gratitude to all those people who have made this dissertation possible and because of whom my graduate experience has been one that I will cherish forever. My deepest gratitude is to my advisor, Prof. S.P. Ghrera. I have been amazingly fortunate to have an advisor who gave me the freedom to explore on my own, and at the same time the guidance to recover when my steps faltered. His patience and support helped me overcome many crisis situations and finish this dissertation.

DPMC members, Prof. G. Singh, Dr. R Bhat and Dr. P. Khokhar have been always there to listen and give advices. I am deeply grateful to all of them for the long discussions that helped me sort out the technical details of my work. I am also thankful to Prof. G. Singh for encouraging the use of proper timeline and for carefully reading and commenting on writing this manuscript.

I am grateful to my colleagues for their encouragement and practical advices. I am also thankful to them for reading my reports, commenting on my views and helping me understand and enrich my ideas.

My sincere thanks to the undergraduate students and staff of the University, for their various forms of support during my graduate study.

Many friends have helped me stay sane through these difficult years. Their support

---

and care helped me overcome setbacks and stay focused on my graduate study. I greatly value their friendship and I deeply appreciate their belief in me.

Most importantly, none of this would have been possible without the love and patience of my family, has been a constant source of love, concern, support and strength all these years.

Finally, I appreciate the financial and academic support of the Jaypee University of Information technology.



(Suman Saha)

Dept. Computer Science

Jaypee University of Information Technology

Waknaghat, Solan, H.P.

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Introduction . . . . .	2
1.2	Complex networks . . . . .	4
1.2.1	Communication graphs . . . . .	4
1.2.2	Collaboration networks . . . . .	5
1.2.3	The www and blogs . . . . .	6
1.3	Properties of complex networks . . . . .	6
1.3.1	Diameter . . . . .	7
1.3.2	Navigability . . . . .	8
1.3.3	Clustering coefficients . . . . .	8
1.3.4	Degree distributions . . . . .	9
1.4	Network community detection . . . . .	10
1.4.1	Evaluation measures of network communities . . . . .	10
1.4.2	Community detection algorithms . . . . .	12
1.5	Thesis road map . . . . .	17
1.5.1	Network community detection on metric space . . . . .	18

1.5.2	Nearest Neighbor search in Complex Network for Community De-	
	tection . . . . .	18
1.5.3	Low rank approximations for network community detection . . . . .	19
1.5.4	Optimal evaluation of network community . . . . .	20
1.6	Conclusion . . . . .	21
<b>2</b>	<b>Network Community Detection on Metric Space</b>	<b>23</b>
2.1	Introduction . . . . .	24
2.2	Network community detection . . . . .	26
2.2.1	Popular algorithms . . . . .	26
2.2.2	Observations and motivations . . . . .	28
2.3	Graph to metric space transformation . . . . .	30
2.3.1	Graph to metric space algorithm . . . . .	31
2.4	Community detection on induced metric space . . . . .	33
2.4.1	k-partitioning . . . . .	34
2.4.2	Initialization . . . . .	34
2.4.3	Convergence . . . . .	35
2.4.4	Data complexity . . . . .	37
2.5	Experiments and results . . . . .	39
2.5.1	Experimental designs . . . . .	39
2.5.2	Performance indicator . . . . .	40
2.5.3	Datasets . . . . .	40
2.5.4	Computational results . . . . .	40
2.5.5	Parameter settings . . . . .	44
2.5.6	Results analysis and achievements . . . . .	44

2.6	Conclusions . . . . .	45
<b>3</b>	<b>Nearest Neighbor search in the Metric Space of Complex Network for Community Detection</b>	<b>47</b>
3.1	Introduction . . . . .	48
3.2	Notion of nearness in complex network . . . . .	50
3.2.1	Definitions . . . . .	50
3.2.2	Nearness in complex network . . . . .	51
3.2.3	Nearness in complex network: . . . . .	52
3.3	Nearest neighbor search on complex network using metric tree . . . . .	54
3.3.1	Metric-tree . . . . .	54
3.3.2	Nearest Neighbor search algorithm using M-Tree . . . . .	55
3.4	Nearest neighbor search on complex network using locality sensitive hashing	56
3.4.1	Approximate nearest neighbor . . . . .	56
3.4.2	Locality Sensitive Hashing (LSH) . . . . .	57
3.4.3	Locality sensitive hash function for complex network . . . . .	57
3.5	Proposed community detection based on nearest neighbor . . . . .	58
3.5.1	Distance based community detection . . . . .	59
3.5.2	Proposed algorithm for network community detection using nearest neighbor search . . . . .	60
3.5.3	Complexity and convergence . . . . .	60
3.6	Experiments and results . . . . .	62
3.6.1	Experimental Designs . . . . .	62
3.6.2	Datasets . . . . .	63
3.6.3	Exp1: Experiment with nearness measure . . . . .	63



3.6.4	Exp2: Experiment on approximation . . . . .	64
3.6.5	Exp3: Experiment to evaluate proposed algorithm . . . . .	65
3.6.6	Results analysis and achievements . . . . .	70
3.7	Conclusions . . . . .	70
<b>4</b>	<b>Low rank approximations for community detection in large complex networks</b>	<b>73</b>
4.1	Introduction . . . . .	74
4.2	Low rank approximation of matrices . . . . .	76
4.3	Network community detection using low rank approximation . . . . .	78
4.3.1	Distance matrix of complex network . . . . .	78
4.3.2	Nystrom approximation . . . . .	79
4.3.3	Random Sketching . . . . .	80
4.3.4	Approximate community detection . . . . .	81
4.4	Experiments and results . . . . .	84
4.4.1	Experimental designs . . . . .	84
4.4.2	Computational results . . . . .	84
4.4.3	Results analysis and achievements . . . . .	85
4.5	Conclusions . . . . .	86
<b>5</b>	<b>Optimal consensus-community detection for complex network</b>	<b>89</b>
5.1	Introduction . . . . .	90
5.2	Rough Set Theory . . . . .	92
5.2.1	Introduction to Rough Set Theory . . . . .	92
5.3	Optimal community detection using rough set . . . . .	95
5.3.1	Description of method . . . . .	96

---

5.3.2	Consensus community determination . . . . .	97
5.3.3	Significance of input algorithms . . . . .	99
5.3.4	Relation between discernibility and diversity of input communities .	101
5.3.5	Achievements . . . . .	102
5.4	Theoretical results . . . . .	102
5.5	Experiments and results . . . . .	105
5.5.1	Experimental Designs . . . . .	105
5.5.2	Experimental setup to compare consensus community detection . .	105
5.5.3	Experiment on optimal community structure estimation . . . . .	106
5.5.4	Results analysis and achievements . . . . .	106
5.6	Conclusion . . . . .	107
<b>6</b>	<b>Conclusion</b>	<b>111</b>

# List of Tables

2.1	Algorithms for network community detection and their complexities . . . .	29
2.2	Complex network datasets and values of their parameters . . . . .	41
2.3	Comparison of our approaches with other best methods in terms of con- ductance, the number inside the brackets denotes the algorithm of the group	42
2.4	Comparison of our approaches with other best methods in terms of modu- larity, the number inside the brackets denotes the algorithm of the group .	43
2.5	Comparison of our approaches with other best methods in terms of time(second) . . . . .	43
3.1	Exp1: Experiment with nearness measure . . . . .	64
3.2	Exp2: Experiment on approximation . . . . .	66
3.3	Comparison of our approaches with other best methods in terms of modularity	68
3.4	Comparison of our approaches with other best methods in terms of time .	69
4.1	Comparison of our approaches with other best methods in terms of modularity	85
4.2	Comparison of our approaches with other best methods in terms of time .	86
5.1	Expressing $P$ by lower approximation and upper approximation of other partitions . . . . .	98

5.2	Experiment on optimal community structure estimation . . . . .	108
5.3	Summary of results . . . . .	109

# List of Figures

1.1	An example of communities in complex network . . . . .	3
2.1	Block diagram of k-central algorithm for complex network via metric space transformation . . . . .	36
3.1	Block diagram of k-central algorithm using approximate nearest neighbor search with metric tree or locality sensitive hashing . . . . .	61
4.1	Block diagram of approximate k-central algorithm using om lowrank approximation with Nystrom sampling or random sketching . . . . .	83
5.1	Block diagram of meta community detection algorithm using rough set theory and optimization . . . . .	100



# Chapter 1

## Introduction

Complex Networks acquired a huge popularity and represent one of the most important social and Computer Science phenomena of these years. Analysis of complex network involves several methods ranging from statistical analysis of network features, to graph-theoretical models and finally machine learning techniques, from a quantitative and qualitative perspective. The origin, distribution and sheer size of the data involved makes each of them quite inapplicable to the required scale. The aim of this thesis is development, analysis and applications of complex network under the constrained of approximate data and the distance based model. A comprehensive study of the process of mining information from large complex networks and analyzing the structure of the networks using synopsis of real network is the objective of the work. Various data approximation methods like, sampling and sketching are considered here for creating data synopsis of social media. New methods will be developed and their effectiveness will be assessed against relevant data synopsis.

## 1.1 Introduction

With the rise of on-line networking communities like Facebook and Twitter along with the popularization of the notions of "six degrees of separation" [42] and the "Kevin Bacon game" [7], the complex network gain lot of research interest in recent years. A complex network is simply a structure consisting of entities embedded in a social context, with a relationship among those entities that represents interaction, collaboration, or influence between entities. (One can consider a variety of types of this relationship: the mutual declaration of friendship, an email sent from one person to the other, the co-authorship of a scientific paper, and so forth.) [11] A large number of examples are available for real life complex networks like, citation graphs: (directional) edge  $(X,Y)$  exists if paper  $X$  cites paper  $Y$ , collaboration graphs: (bidirectional) edge  $(X,Y)$  if person  $X$  worked with person  $Y$ , semantic graphs: dictionaries, thesaurus; edge  $(X,Y)$  exists if word  $X$  is associated with word  $Y$ , biological graphs: edge  $(X,Y)$  exists if process  $X$  is related to process  $Y$  (e.g. protein interactions, predators, food webs), communication graphs: computer networks or telephone networks, news graphs: relationship of events, words, or people in the news, the Internet and the worldwide web is a directed graph and social networks(facebook, twitter, google+,...) to name a few.

complex networks present fascinating patterns and properties [53] like, the degree distribution follow power-law, the average distance between the nodes of the network is short (the small-world phenomenon), unlike a random graphs it has high clustering coefficients [31].

Community detection in complex network aims to identify the modules and, possibly, their hierarchical organization, by only using the information encoded in the graph topology [54, 74]. First attempt dates back to 1955 by Weiss and Jacobson searching for

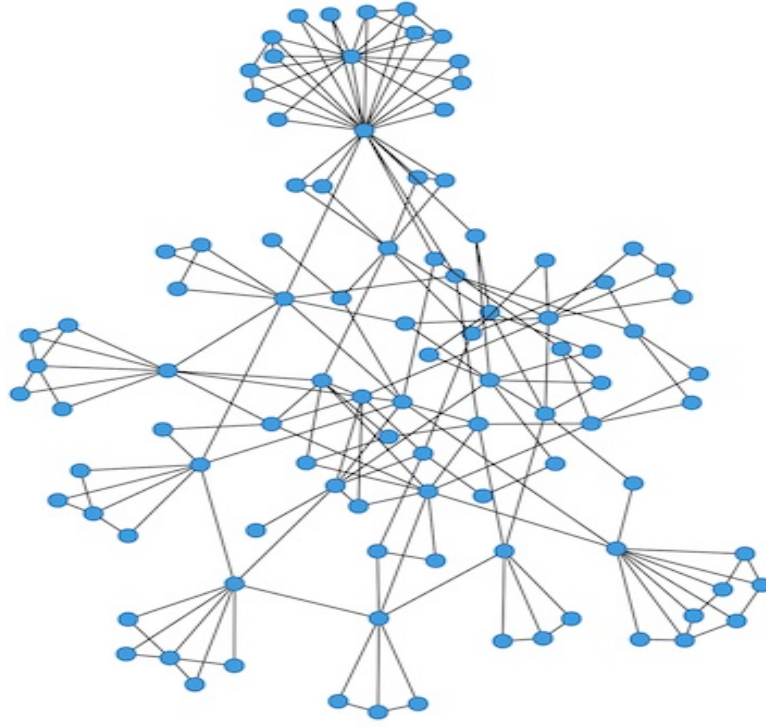


Figure 1.1: An example of communities in complex network

work groups within a government agency [78]. Network community detection has wide variety of application on several domains. Social communities have been studied for a long time. In biology - protein-protein interaction networks, communities are likely to group proteins having the same specific function within the cell. In World Wide Web the communities correspond to groups of pages dealing with the same or related topics. In metabolic networks communities may be related to functional modules such as cycles and pathways whereas, in food webs they may identify compartments. A detailed of the cases are given in later section of this chapter.

The observations and motivation to our approaches in this thesis are briefed in this paragraph. Network community detection is not easy NP-Hard like data clustering due to

the lack of good heuristics. Both, Graph traversal based methods and spectral methods are computationally overloaded due to the verification of objective function value, required to guide next iteration. Rich literature of clustering are not suitable for graph data. These observations motivates us for a transformation from complex network to Metric Space. But the metrics developed so far on graph (like Shortest path, Jaccard similarity and Euclidean distance between adjacency vectors) are less successful for network community detection in terms of conductance and modularity so there is a need for development of a good metric which works better on complex network.

## **1.2 Complex networks**

Complex network analysis is a large and growing body of research on the measurement and analysis of relational structure. The complex network field is an interdisciplinary research program which seeks to predict the structure of relationships among social and natural entities, as well as the impact of said structure on other social phenomena. The elements of this research are built around the concepts and methods for the measurement, representation, and analysis of network structure. These techniques referred to as the methods of complex network analysis are applicable to a wide range of domains, ranging from the analysis of concepts within mental models to the study of war between nations. Several examples of real life complex networks are discussed in the following subsections.

### **1.2.1 Communication graphs**

One automatic way to extract complex networks is to define friendship in terms of communication and extract edges from records of that communication. Call graphs are one natural basis for deriving a communication-based network. Here nodes are telephone

numbers, and a (directed) edge is recorded from  $u$  to  $v$ , with a timestamp  $t$ , if a call is placed from  $u$  to  $v$  at time  $t$ . There are some problems of node conflation here, as more than one person may be associated with the same phone number, and call data are not always readily available outside of telecommunications companies. One can also use email networks to record social interactions, analogously to call graphs. Here there are issues of privacy, so most studies have taken place on relatively limited email datasets that are not publicly available. An interesting recent development on this front was the late 2003 release by the Federal Energy Regulatory Commission of around 1.5 million emails from the top executives at Enron, containing about 150 Enron users and about 2000 total people.

### 1.2.2 Collaboration networks

Another source of data on interactions is collaboration networks: an (undirected) edge exists between  $u$  and  $v$  if they have collaborated on a mutual project. One can consider many different types of projects, but the most-studied collaboration networks are based on the Hollywood-actor collaboration graph, where actors who jointly appear in a movie are connected by an edge, and the academic collaboration graph, where researchers who coauthor an academic paper are connected by an edge. (Citation networks, in which the nodes are papers connected by directed edges from each paper to the papers that it cites, are not complex networks, in that their edges point only backwards in time, and do not really represent a social relationship between their endpoints.) A collaboration network as we have described it here is a derived structure that loses some of the information present in the original data. For a lossless representation of a collaboration network, a set of collaborations can be represented by a bipartite graph of people and projects, where



a person  $u$  is connected to a project  $p$  if and only if  $u$  worked on  $p$ . The collaboration graph as described above looks only at the people in the bipartite graph, connecting  $u$  to  $v$  exactly if there is a length-two path connecting  $u$  and  $v$  in the bipartite graph.

### 1.2.3 The www and blogs

The web itself has the link structure to form a complex network where the pages are the node of the network and links are directed edges. Particular types of web pages, though, have much more purely social link structure. A blog (an abbreviation of web-log) is an on-line diary, often updated daily (or even hourly), typically containing reports on the user's personal life, reactions to world events, and commentary on other blogs. Links on blog pages are often to other blogs read by the author, or to the blogger's friends, and thus the link structure within blogging communities is an essentially social relationship. Thus complex networks can be derived from blog-to-blog links on a blogging community website.

## 1.3 Properties of complex networks

Researchers have concentrated particularly on a few properties that seem to be common to many networks: the small-world property, power-law degree distributions, and network transitivity. Small world effect is the finding that the average distance between vertices in a network is short, usually scaling logarithmically with the total number  $n$  of vertices. Right-skewed degree distribution is another property that many networks possess. The degree of a vertex in a network is the number of other vertices to which it is connected, and one finds that

there are typically many vertices in a network with low degree and a small number

with high degree, the precise distribution follow a power-law or exponential form. A third property that many networks have in common is network transitivity, which is the property that two vertices that are both neighbors of the same third vertex have a heightened probability of also being neighbors of each other. In the language of complex networks, two of your friends will have a greater probability of knowing one another than will two people chosen at random from the population, on account of their common acquaintance with you. Now that we have defined the social-network models most relevant for the remainder of this thesis, we turn to a brief survey of some important properties of complex networks that have been discussed in the literature. Mathematically, a complex network is a graph  $G = (V, E)$ , where the nodes represent entities, and an edge  $(u, v) \in E$  denotes some type of relationship between the entities  $u$  and  $v$ . We use graph-theoretic terminology in this thesis, but readers whose background is in a different field can mentally translate node into vertex, actor, or site, and edge into arc, tie, link, or bond.

### 1.3.1 Diameter

There are short chains of friends that connect a large fraction of pairs of people in a complex network. That is, complex networks have a small diameter in the graph-theoretic sense of the longest shortest path. Unfortunately, the literature on complex networks tends to use the word "diameter" ambiguously in reference to at least four different quantities: (1) the longest shortest-path length, which is the true graph theoretic diameter but which is infinite in disconnected networks, (2) the longest shortest-path length between connected nodes, which is always finite but which cannot distinguish the complete graph from a graph with a solitary edge, (3) the average shortest-path length, and (4) the average shortest-path length between connected nodes. A wide variety of other intrigu-

ing networks have been shown to have small average shortest-path length, of which we highlight the world-wide web, for which the average (directed) shortest-path length was estimated to be under twenty [29].

### **1.3.2 Navigability**

A further characteristic observed in real complex networks is that they are navigable small worlds: not only do there exist short paths connecting most pairs of people, but using only local information and some knowledge of global structure, for example, each person  $u$  in the network might know the geographic layout of the United States and the geographic locations of only some target individual and each of  $u$ 's own friends, the people in the network are able to construct short paths to the target. Although no rigorous theoretical analysis of it has been given, we would be remiss if we proceeded without mentioning the small-world model defined by Watts, Dodds, and Newman, in which navigation is also possible. Their model is based upon multiple hierarchies (geography, occupation, hobbies, etc.) into which people fall, and a greedy algorithm that attempts to get closer to the target in any dimension at every step. Simulations have shown this algorithm and model to allow navigation, but no theoretical results have been established.

### **1.3.3 Clustering coefficients**

The clustering coefficient of complex networks is much higher than is predicted in a random graph. Informally, the clustering coefficient measures the probability that two people who have a common friend will themselves be friends or not, in graph theoretic terms, the fraction of triangles in the graph that are closed. There are a handful of different ways to measure the clustering coefficient in a graph formally; here we mention

just one, for concreteness. The clustering coefficient for a node  $u \in V$  of a graph  $G = (V, E)$  is the fraction of edges that exist within the neighborhood of  $u$ , i.e., between two nodes adjacent to  $u$ . The clustering coefficient of the entire network is the average clustering coefficient taken over all nodes in the graph. (Other formalizations compute the probability that the third edge  $\{v, w\}$  exists when we choose an ordered triple  $(u, v, w)$  such that  $\{u, v\}, \{u, w\} \in E$  is chosen uniformly at random from the set of all such triples; the formalization described above gives relatively less weight to high-degree nodes than in this alternative approach.) Typical complex networks have clustering coefficients on the order of 10 to 1, which is orders of magnitude greater than the clustering coefficient that a random graph.

### 1.3.4 Degree distributions

The small-world models that we have discussed up until now have contained essentially homogeneous nodes; variations among nodes are relatively minor. In particular, the degree distribution of the network, that is, the proportion of nodes in the network who have a particular degree  $\delta$  (i.e., the fraction of people with exactly  $\delta$  friends), for every degree  $\delta > 0$ , has been essentially constant in the random graphs of Erdos and Rinyi, Watts and Strogatz, and Kleinberg. Real social networks, however, show a wide heterogeneity in the popularity of their nodes, and their degree distributions are extremely poorly predicted by any of the models discussed above. In a typical complex network, the proportion of nodes with degree at least  $\delta$  is reasonably well approximated by the power-law distribution  $P(\delta)$  varies with  $\delta^{-\beta}$ , for a constant  $\beta > 0$ , usually where  $\beta$  ranges between 2.1-2.5. (Others have referred to networks exhibiting a power-law degree distribution as scale-free networks, or as exhibiting Pareto, heavy-tailed, or Zipfian degree distributions.) This

phenomenon was noted for the degree distribution of the world-wide web, and has also been observed in other complex networks.

## 1.4 Network community detection

Community detection in real networks aims to capture the structural organization of the network using the connectivity information as input [54, 74]. Early work on this domain was attempted by Weiss and Jacobson while searching for a work group within a government agency [78]. Most of the methods developed for network community detection are based on a two-step approach. The first step is specifying a quality measure (evaluation measure, objective function) that quantifies the desired properties of communities and the second step is applying an algorithmic techniques to assign the nodes of graph into communities by optimizing the objective function. Several measures for quantifying the quality of communities have been proposed, they mostly consider that communities are set of nodes with many edges between them and few connections with nodes of different communities. Some of the community evaluation measures are described in the next subsection.

### 1.4.1 Evaluation measures of network communities

Several measures for quantifying the quality of communities have been proposed, they mostly consider that communities are set of nodes with many edges between them and few connections with nodes of different communities. Some of the community evaluation measures are described in the next subsection.

**Modularity:** The notion of modularity is the most popular for the network community detection purpose. The modularity index assigns high scores to communities whose inter-



nal edges are more than that expected in a random-network model which preserves the degree distribution of the given network.

**Internal Density:** Density is defined by the number of edges ( $m_s$ ) in subset  $S$  divided by the total number of possible edges between all nodes ( $n_s(n_s - 1)/2$ ). The "2" is there to cancel out duplicated edges. Internal Density =  $m_s/(n_s(n_s - 1)/2)$

**Edges Inside:** Somewhat useless by itself since it is not related to any other attributes of subset  $S$ . The total number of edges ( $m_s$ ) present in subset  $S$ . Edges Inside =  $m_s$

**Average degree:** The average internal degree across all nodes ( $n_s$ ) in subset  $S$ . Average Degree =  $2m_s/n_s$

**Fraction over median degree:** Determines the number of nodes that have an internal degree greater than the median degree of nodes in Subset  $S$ .

**Triangle Participation Ratio:** The best measure for density, cohesiveness, and clustering within the goodness scales. Robust under random and expand perturbations. The fraction of nodes in  $S$  that belong to a triad. TPR = (number of nodes belonging to a triad)/ $n$ .

**Expansion:** This measure of separability gives the average of number of external connections ( $c_s$ ) per node ( $n_s$ ) in subset  $S$  has with graph  $G$ . It can be thought of external degree. Expansion =  $c_s/(n_s(n - n_s))$ .

**Cut Ratio:** This metric is a measure of separability and can be thought of as external density. It is the fraction of external edges ( $c_s$ ) of subset  $S$  out of the total number of possible edges in graph  $G$ .

**Conductance:** Ratio of edges inside the cluster to the number of edges leaving the cluster (captures surface area to volume). Measures best in separability (goodness scale) measures well separated non-overlapping communities. Robust under node swap and shrink perturbation. Community like sets of nodes have lower conductance.

**Normalized Cut:** Represents how well subset  $S$  is separated from graph  $G$ . It sums up the fraction of external edges over all edges in subset  $S$  (conductance) with the fraction of external edges over all non-community edges.

**Maximum Out Degree Fraction:** This metric first finds the fraction of external connections to internal connections for each node ( $n_s$ ) in  $S$ . It then returns the fraction with the highest value.

**Average Out Degree Fraction:** The sum of the individual fraction of edges outside of the community over the total connections of a node in subset  $S$ . It is then divided by the total number of nodes ( $n_s$ ) in subset  $S$ .

**Flake Out Degree Fraction:** This is a fraction of the number of nodes that have fewer internal connections than external connections to the number of nodes ( $n_s$ ) in subset  $S$ .

There are several other measures of quality determination for network community. However, the most widely used measures are modularity and conductance. Majority of the algorithms are developed using either of the measure as their optimization criteria.

### 1.4.2 Community detection algorithms

Identifying communities in a network is an important issue for many real-world applications in various scientific fields. Over the years, many methods have been devised to provide efficient community discovery algorithms. As the spectrum is wide, building taxonomy of solutions is not easy. They can be classified in different ways, and depending on the selected criteria, one algorithm can belong to more than one category. Here, we choose to focus on the process implemented by the algorithms. We consider this as their main characteristic, since it directly affects the nature of the detected communities. As a result, we group the algorithms in six different categories. In this section, we describe

these categories and the representative set of algorithms we selected.

### **Centrality based algorithms**

The algorithms based on link-centrality measures rely on a hierarchical divisive approach. Initially the whole network is seen as a single community, i.e. all nodes are in the same community. The most central links are then repeatedly removed. The underlying assumption is that these particular links are located between the communities. After a few steps, the network is split in several components which can be considered as communities in the initial network. Iterating the process, one can split each discovered community again, resulting in a finer community structure. This eventually leads to a network in which each node is isolated, and therefore constitutes its own community. By considering the communities detected at each step of the process, one obtains a hierarchy of community structures. The choice of the best one is generally performed using a measure estimating the quality of community structures, such as the modularity [52]. Algorithms of this category differ in the way they select the links to be removed. The first and most known algorithm using this approach was proposed by Newman [55], and relies on the edge-betweenness measure. It estimates the centrality of a link by considering the proportion of shortest paths going through it in the whole network. As the complexity of this algorithm is high, it is not well suited for very large networks. Radicchi et al. [66] proposed a variation based on link transitivity instead of edge-betweenness. This measure is defined as the number of triangles to which a given link belongs, divided by the number of triangles that might potentially include it. Its lower complexity makes it more appropriate for large networks.

## Modularity Optimization Algorithms

Modularity is a prominent measure of the quality of a community structure introduced by Newman and Girvan [34]. It measures internal connectivity of identified communities with reference to a randomized null model with the same degree distribution. Modularity optimization algorithms try to find the best community structure in terms of modularity. They diverge on the optimization process they are based on. As this approach is very influential in the community detection literature we consider three algorithms for investigation. FastGreedy developed by Newman et al. [55] relies on a greedy optimization method applied to a hierarchical agglomerative approach. The agglomerative approach is symmetrical to the divisive one described in the previous subsection. In the initial state, each node constitutes its own community. The algorithm merges those communities step by step until only one remains, containing all nodes. The greedy principle is applied at each step, by considering the largest increase (or smallest decrease) in modularity as the merging criterion. Because of its hierarchical nature, FG produces a hierarchy of community structures like the divisive approaches. The best one is selected by comparing their modularity values. Another optimization algorithm [70] is an improvement of fastgreedy, introducing a two-phase hierarchical agglomerative approach. During the first phase, the algorithm applies a greedy optimization to identify the communities. During the second phase, it builds a new network whose nodes are the communities found during the first phase. The intra-community links are represented by self-loops, whereas the inter-community links are aggregated and represented as links between the new nodes. The process is repeated on this new network, and stops when only one community remains. Spinglass by Reichardt and Bornholdt [68] relies on an analogy between a very popular statistical mechanic model called Potts spin glass, and the community structure.

It applies the simulated annealing optimization technique on this model to optimize the modularity.

### **Spectral Algorithms**

Spectral algorithms take advantage of various matrix representations of networks. Classic spectral graph partitioning techniques focus on the eigenvectors of the Laplacian matrix. They were designed to find the partition minimizing the links lying in-between node groups. However, these methods were designed for slightly different contexts (e.g. user-specified number of communities). For real-world complex networks, the community number is unknown. Thus, these methods are not efficient in our case. The methods we selected are variants adapted to complex networks analysis. Leading Eigenvector algorithm [81] applies the classic graph partitioning approach, but to a so-called modularity matrix instead of the Laplacian. Doing so, it performs an optimization of the modularity instead of the objective measures used in classic graph partitioning, such as the minimal cut. Commfind is developed by Donetti and Munoz [24]. It combines the analysis of the Laplacian matrix eigenvectors used in classic graph partitioning with a cluster analysis step. Instead of using the best eigenvector to iteratively perform bisections of the network, it takes advantage of the  $\gamma$ L best ones. Communities are obtained by a cluster analysis of the projected nodes.

### **Graph traversal based algorithms**

Several algorithms use random walks in various ways to partition the network into communities. We retain two of them in our comparisons. Walktrap (WT) by Pons and Latapy [65] uses a hierarchical agglomerative method like fast greedy but with a different merging criterion, which relies on the modularity measure, WT uses a node-to- node dis-

tance measure to identify the closest communities. This distance is based on the concept of random-walk. If two nodes are in the same community, the probability to get to a third one located in the same community through a random walk should not be very different for both of them. The distance is constructed by summing these differences over all nodes, with a correction for degree.

### **Information based algorithms**

Information-Based algorithms use tools derived from the information theory to estimate the best partition of the network. The main idea of those approaches is to take advantage of the community structure in order to represent the network using less information than that encoded in the full adjacency matrix. We selected two algorithms from this category. Infomod and Infomap was proposed by Rosvall and Bergstorm [70]. It is based on a simplified representation of the network focusing on the community structure: a community matrix and a membership vector. The former is an adjacency matrix defined at the level of the communities (instead of the nodes), and the latter associates each node to a community. The authors use the mutual information measure to quantify the amount of information from the original network contained in the simplified representation. They obtain the best partition by considering the representation associated to the maximal mutual information. The community structure is represented through a two-level nomenclature based on Huffman coding in Infomap: one to distinguish communities in the network and the other to distinguish nodes in a community. The problem of finding the best partition is expressed as minimizing the quantity of information needed to represent some random walk in the network using this nomenclature. With a partition containing few inter-community links, the walker will probably stay longer inside communities, therefore only the second level will be needed to describe its path, leading to a

compact representation. The authors optimize their criterion using simulated annealing.

### **Other algorithms**

A number of algorithms do not fit in the previously described approaches. We selected the Label Propagation (LP) algorithm by Raghavan et al. [67], which uses the concept of node neighborhood and simulates the diffusion of some information in the network to identify communities. Initially, each node is labeled with a unique value. Then an iterative process takes place, where each node takes the label which is the most spread in its neighborhood (ties are broken randomly). This process goes on until convergence, i.e. each node has the majority label of its neighbors. Communities are then obtained by considering groups of nodes with the same label. By construction, one node has more neighbors in its community than in the others.

There are more algorithms developed to solve network community detection problem a complete list can be obtained in several survey articles [29, 46, 80]. Some interesting recent articles are [1, 4, 22, 23, 44, 58, 81].

## **1.5 Thesis road map**

Aim of this work is the development of fast and accurate algorithms for community detection of large network. Competitiveness of the several approximation methods being analyzed with respect to their modularity value and computational complexity. The main contribution of the thesis is explained via the following chapters: 1) Chapter 2: Network community detection on metric space 2) Chapter 3: Nearest Neighbor search in Complex Network for Community Detection, 3) Chapter 4: Low rank approximations for community detection of very large networks, and 4) Chapter 5: Optimal evaluation of

network community.

### 1.5.1 Network community detection on metric space

Nodes of the graph does not lies on a metric space. e.g. edges does not reflect the Euclidean distance between the nodes. In this work, we have tried to develop the notion of similarity among the vertices using some new matrices derived from adjacency matrix and degree matrix of the graph. Let  $A$  be the adjacency matrix and  $D$  the degree matrix of the graph  $G = (V, E)$  then Laplacian  $L = D - A$ . The standard Euclidean distance and spherical distance define over the matrices above failed to capture similarity information among the nodes of  $G$ . Algorithms developed based on shortest path or Jaccard similarity are computationally inefficient and have less success in terms of standard evaluation criteria(like, conductance and modularity)

### 1.5.2 Nearest Neighbor search in Complex Network for Community Detection

The nearest neighbor search is a basic computational tool used extensively in almost research domains of computer science specially when dealing with large amount of data. However, the use of the nearest neighbor search is restricted for the purpose of algorithmic development by the existence of the notion of nearness among the data points. The recent trend of research is on large complex networks and their structural analysis. In complex network, nodes represent entities and edges represent any kind of relation between entities. Community detection in complex network is an important problem of much interest. In general, a community detection algorithm represents an objective function and captures the communities by optimizing it to extract the interesting communities for the user. In



this article, we have studied the nearest neighbor search problem in complex network via the development of a suitable notion of nearness. Initially, we have studied and analyzed the exact nearest neighbor search using the metric tree on proposed metric space constructed from complex network. The approximate nearest neighbor search problem is studied using locality sensitive hashing. For evaluation of the proposed nearest neighbor search on complex network, we applied it in community detection problem. The results obtained using our methods are very competitive with most of the well known algorithms exists in the literature and this is verified on a collection of real networks. On the other-hand, it can be observed that the time taken by our algorithm is quite less compared to popular methods. Achievements of using Approximate Nearest Neighbor search on complex networks are 1) Compute sub-linear time metric query,  $O(\text{Log}(n))$  and 2) it hardly degrade the results of actual algorithms.

### 1.5.3 Low rank approximations for network community detection

In this work we will explore the two low rank approximation methods, Nystrom sampling and random sketching. The Nystrom methods approximate any SPSD matrix in terms of a subset of its columns. Specifically, given an  $m \times m$  SPSD matrix  $A$ , they require sampling  $c(< m)$  columns of  $A$  to construct an  $m \times c$  matrix  $C$ . We always assume that  $C$  consists of the first  $c$  columns of  $A$  without loss of generality. We partition  $A$  and  $C$  as

$$A = \begin{pmatrix} W & A_{21}^T \\ A_{21} & A_{22} \end{pmatrix} \text{ and } C = \begin{pmatrix} W \\ A_{21} \end{pmatrix}$$

where  $W$  and  $A_{21}$  are of size  $c \times c$  and  $m - c \times c$  respectively [43]. Since the running time complexity of SVD on  $W$  is in  $O(kl^2)$  and matrix multiplication with  $C$  takes  $O(kln)$ , the total complexity of the Nystrom approximation computation is in  $O(kln)$  [2, 25].

Random sketching is described by the use of Johnson-Lindenstrauss theorem, that there is a distribution over random linear mappings  $A : R^d \rightarrow R^k, (k < d)$  such that for any vector  $x$  we have  $\|Ax\| = (1 \pm \epsilon)\|x\|$  with probability  $1 - e^{-Ck\epsilon^2}$ . The Johnson-Lindenstrauss theorems [40] for sketches ensures the preservation of distance during lower dimensional approximation and provide minimum lower bound of approximation.

Achievements of low rank methods are 1) they are very fast to compute massive data and 2) results approximated with moderate lower bound.

#### 1.5.4 Optimal evaluation of network community

In this chapter, we compare communities of complex networks determined by several algorithms using Rough Sets theory and tried to combine them to obtain an optimally community structure of the network.

Rough set (developed by Polish mathematician Zdzislaw I. Pawlak in 1991) is a formal approximation of a concept in terms of a pair of sets, the lower and the upper approximation derived from the available instances [64]. Formally, an information system is defined as  $I = (U, A, V, f)$ , where  $U$  is a nonempty and finite set of objects, called the universe;  $A$  is a nonempty and finite set of attributes;  $V$  is the union of attribute domains, i.e.,  $V = \cup V_a$ , where  $V_a$  denotes the domain for each attribute  $a \in A$ ; and  $f$  is an information function which associates a unique value of each attribute with every object belonging to  $U$ . An arbitrary attribute set  $B \subseteq A$  determines a binary relation  $IND(B)$ , called indiscernibility relation.  $IND(B) = \{ \langle x, y \rangle \in U \times U \mid \forall b \in B, f(x, b) = f(y, b) \}$ . Equivalence

partition denoted as  $U/B$  and blocks are written as  $[x]_B = \{y \in U \mid \langle x, y \rangle \in IND(B)\}$ . Let  $C \subseteq U$  be a concept, the lower and upper approximations with respect to  $B \subseteq A$  are denoted as  $\underline{B}(X) = \{x \in U \mid [x]_B \subseteq X\}$  &  $\overline{B}(X) = \{x \in U \mid [x]_B \cap X \neq \Phi\}$

## 1.6 Conclusion

In this work the fast and optimal algorithms for community detection on large network is described and analyzed. We demonstrated and analyzed a new approaches to network community detection via metric space induced by the complex network. The interesting problem of the nearest neighbor within the nodes of a complex networks are studied and applied for community detection. We have used geometric framework for network community detection instead of traditional graph theoretic approach or spectral methods. We presented the efficient computation of network community detection using low rank approximation. Our techniques can be applied for quick analysis of very large complex network. Finally the theoretical upper bound of network community detection algorithms is analyzed using the notion of data complexity and tried to estimate the optimal bound by heterogeneous combination of communities. Competitiveness of the several approximation methods being analyzed with respect to their modularity value and computational complexity and found very promising with respect to time gain.

## Chapter 2

# Network Community Detection on Metric Space

Community detection in complex network is an important problem of much interest in recent years. In general, a community detection algorithm chooses an objective function and captures the communities of the network by optimizing the objective function and then one uses various heuristics to solve the optimization problem to extract the interesting communities for the user. In this chapter we have demonstrated the procedure to transform a graph into points of a metric space and developed the methods of community detection with the help of metric defined for pair of points. We have also studied and analyzed the community structure of the network therein. The results obtained with our approach are very competitive with most of the well known algorithms in the literature and this is justified over the large collection of datasets. On the other-hand, it can be observed that time taken by our algorithm is quite less compared to other methods and justify the theoretical findings.

## 2.1 Introduction

The rise of on-line networking communities in real-world graphs such as large social networks, web graphs, and biological networks have initiated the important direction of network community detection [11, 31, 56, 66]. A network community (also known as a module or cluster) is typically a group of nodes with more interactions among its members than the remaining part of the network [29, 74, 78]. To extract such group of nodes of a network one typically selects an objective function that captures the intuition of a community as a set of nodes with better internal connectivity than external [49, 54]. The objective is generally NP-hard to optimize [54, 74], heuristics [18, 19] or approximation algorithms [74] are used in practice to find sets of nodes that approximately optimize the objective function, which is interpreted as real communities.

Another important approach is to define communities as output of an algorithm which converges automatically, with some intuitive hope to extract good communities [46, 67]. Identified communities have some different importance in different domain. In social network community means an organizational unit, in biochemical network a functional unit, in collaboration network a scientific discipline and so. on. [80].

Our observations regarding the development of network community detection algorithms are as follows: 1) the network community detection is not easy NP-Hard like data clustering due to the lack of good heuristics, 2) both, graph traversal based methods and spectral methods are computationally overloaded due to the verification of objective function value, which is required to guide next iteration, and 3) rich literature of clustering are not very suitable for graph data.

Some methods are available for network community detection which tries to develop similarity or distance function among the nodes of a complex network and use that simi-

larity or distance for partitioning the network [8, 9, 12, 26, 47, 50, 65]. Most of the methods of community detection, based on similarity or distance, mainly use shortest path, Jaccard similarity, set similarity or Euclidean distance and they are less successful for network community detection in terms of conductance and modularity. In some cases weighted graph are requirement which is not always obtained naturally in the real networks. Complex networks are characterized by small average path length and high clustering coefficient the way the metric is defined should be able to capture the crucial properties of complex networks. Therefore, we need to create the metric very carefully so that it can explore the underlying community structure of the real life networks.

In this work, we have developed the notion of metric among the nodes using some new matrices derived from modified adjacency matrix of the graph which is flexible over the networks and can be tuned to enhance the structural properties of the network required for community detection. The main contributions of this work include:

- Detailed study of the community detection algorithms.
- Transforming a graph to metric space preserving its structural properties.
- Studying the complex properties of real world networks on induced metric space.
- Developing community detection algorithms on induced metric space.
- Analyzing the results and complexities of the developed algorithms.
- Comparing the community detection algorithms with other existing methods.

The rest of this chapter is organized as follows: Section 2.2 described the state of the art of network community detection literature. In Section 2.3, the problem of transforming a graph into a metric space is discussed and the properties of real complex network is

studied. In Section 3.5.2, the problem of network community detection is formulated and several possible solutions are presented in the induced metric space. Also, the initialization procedures, termination criteria, convergence are discussed in detail. The results of comparison between community detection algorithms are illustrated in Section 2.5. The computational aspects of the proposed framework are also discussed in this section.

## **2.2 Network community detection**

Community detection in real networks aims to capture the structural organization of the network using the connectivity information as input [54, 74]. Early work on this domain was attempted by Weiss and Jacobson while searching for a work group within a government agency [78].

Most of the methods developed for network community detection are based on a two-step approach. The first step is specifying a quality measure (evaluation measure, objective function) that quantifies the desired properties of communities and the second step is applying an algorithmic techniques to assign the nodes of graph into communities by optimizing the objective function.

Several measures for quantifying the quality of communities have been proposed, they mostly consider that communities are set of nodes with many edges between them and few connections with nodes of different communities. Some of the community evaluation measures are already described in the previous chapter.

### **2.2.1 Popular algorithms**

In this subsection we have given a brief list of the algorithms developed for network community detection purposes. The basic approach and the complexity of execution 2.1

is also briefed in this subsection.

- **Fast Greedy Algorithm:** This algorithm was developed by Newman et al. [17, 55]. It is modularity-based and uses a hierarchical agglomerative approach. It is called fastgreedy because it is significantly faster than older algorithms and use greedy method,.
- **Walktrap Algorithm:** This algorithm by Pons and Latapy [65] uses a hierarchical agglomerative method. Here, the distance between two nodes is defined in terms of random walk process. The basic idea is that if two nodes are in the same community, the probability to get to a third node located in the same community through a random walk should not be very different. The distance is constructed by summing these differences over all nodes, with a correction for degree.
- **Eigenvector Algorithm:** This algorithm by Newman [52] is modularity-based, and it uses an optimization method inspired by graph partitioning techniques. It relies on the eigenvectors of a so-called modularity matrix, instead of the graph Laplacian traditionally used in graph partitioning.
- **Label Propagation Algorithm:** This algorithm by Raghavan et al. [67] uses the concept of node neighborhood and the diffusion of information in the network to identify communities. Initially, each node is labeled with a unique value. Then an iterative process takes place, where each node takes the label which is the most spread in its neighborhood. This process goes on until one of several conditions is met, for in-stance no label change. The resulting communities are defined by the last label values.
- **Spinglass Algorithm:** This algorithm by Reichardt and Bornholdt [68] is an opti-



mization method relying on an analogy between the statistical mechanics of complex networks and physical spin glass models

There are more algorithms developed to solve network community detection problem a complete list can be obtained in several survey articles [29, 46, 80]. Some interesting recent articles are [1, 4, 22, 23, 44, 58, 81].

A partial list of algorithms developed for network community detection purpose is tabulated in 2.1. The algorithms are categorized into three main group as spectral (SP), graph traversal based (GT) and semi-definite programming based (SDP). The categories and complexities are also given in the table 2.1.

### **2.2.2 Observations and motivations**

Community detection is an extensively studied research problem of network science. However, a good algorithm for large real network is still in demand for research communities. Two major criteria to be satisfied by the good algorithms are: 1) it must find a partition of the network which is optimal with respect to modularity or conductance, and 2) the algorithm should be computationally efficient on large networks. The notable pitfalls of the existing algorithms are, most of the algorithms developed based on spectral methods on semi-definite programming relies on global optimization and need to compute the costlier functions under the evaluation criteria in each iteration and increase the burden of computation drastically, thus become inefficient for large networks. On the other-hand, graph based algorithms relies on local heuristic method or exhaustive search. The algorithms based on exhaustive search are not suitable for large networks. However the local methods are computationally good but fails achieve close value from optimal modularity for large networks.

Table 2.1: Algorithms for network community detection and their complexities

Author	Ref.	Cat.(No.)	Order
Van Dongen	(Graph clustering, 2000 [76])	GT(1)	$O(nk^2)$ , $k < n$ parameter
Eckmann & Moses	(Curvature, 2002 [28])	GT(2)	$O(mk^2)$
Girvan & Newman	(Modularity, 2002 [34])	SDP(1)	$O(n^2m)$
Zhou & Lipowsky	(Vertex Proximity, 2004 [82])	GT(3)	$O(n^3)$
Reichardt & Bornholdt	(springlass, 2004 [68])	SDP(2)	parameter dependent
Clauset et al.	(fast greedy, 2004 [17])	SDP(3)	$O(n \log_2 n)$
Newman & Girvan	(eigenvector, 2004 [54])	SP(1)	$O(nm^2)$
Wu & Huberman	(linear time, 2004 [79])	GT(4)	$O(n + m)$
Fortunato et al.	(infocentrality, 2004 [30])	SDP	$O(m^3n)$
Radicchi et al.	(Radicchi et al., 2004 [66])	SP(2)	$O(m^4/n^2)$
Donetti & Munoz	(Donetti and Munoz, 2004 [24])	SDP(4)	$O(n^3)$
Guimera et al.	(Simulated Annealing, 2004 [36])	SDP(5)	parameter dependent
Capocci et al.	(Capocci et al., 2004 [10])	SP(3)	$O(n^2)$
Latapy & Pons	(walktrap, 2004 [65])	SP(4)	$O(n^3)$
Duch & Arenas	(Extremal Optimization, 2005 [27])	GT(5)	$O(n^2 \log n)$
Bagrow & Bollt	(Local method, 2005 [6])	SDP(6)	$O(n^3)$
Palla et al.	(overlapping community, 2005 [57])	GT(6)	$O(\exp(n))$
Raghavan et al.	(label propagation, 2007 [67])	GT(7)	$O(n + m)$
Rosvall & Bergstrom	(Infomap, 2008 [71])	SP(5)	$O(m)$
Ronhovde & Nussinov	(Multiresolution community, 2009 [69])	GT(8)	$O(m \beta \log n)$ , $\beta \approx 1.3$

A good alternative is to transform a network to a metric space, where we can achieve good optimality along with automatic convergence, thus lead to less computational burden for large networks, but we need to create the metric very carefully so that it can explore the underlying community structure of the real life networks.

## 2.3 Graph to metric space transformation

In this section we have demonstrated the procedure to transform a graph into points of a metric space and developed the methods of community detection with the help of metric defined for pair of points. We have also studied and analyzed the community structure of the network therein.

As discussed in subsection 2.2.2, the nodes of the graph do not lie on a metric space, e.g. edges does not reflect the Euclidean distance between the nodes. The standard Euclidean distance and spherical distance define over the adjacency or Laplacian matrices above failed to capture similarity information among the nodes of a complex network. On the other-hand, the algorithms developed based on shortest path or Jaccard similarity are computationally inefficient and have less success in terms of standard evaluation criteria(like, conductance and modularity).

In this work, we have tried to develop the notion of similarity among the nodes using some new matrices derived from adjacency matrix and degree matrix of the graph. Let  $A$  be the adjacency matrix and  $D$  the degree matrix of the graph  $G = (V, E)$ . The Laplacian  $L = D - A$ . We have defined two diagonal matrix of same size  $D(\lambda)$  and  $D(\lambda_x)$  where  $\lambda$  is a parameter determine from the given graph and can be optimized from the optimization criteria of the problem under consideration. In  $D(\lambda)$  a fixed optimally determine value is used in the diagonal entries of the matrix  $D$  and in  $D(\lambda_x)$  a variable value also optimally

determine is used in the diagonal entries of the matrix  $D$ . The similarities are defined on matrices  $L_1$  and  $L_2$ , where  $L_1 = D(\lambda) + A$  and  $L_2 = D(\lambda_x) + A$  respectively as spherical similarity among the rows and determine by applying a concave function  $\phi$  over the standard notions of similarities like, Pearson coefficient( $\sigma_{PC}$ ), Spacerman coefficient( $\sigma_{SC}$ ) or Cosine similarity( $\sigma_{CS}$ ).  $\phi(\sigma)()$  must be chosen using the chord condition to obtain a metric.

### 2.3.1 Graph to metric space algorithm

In this subsection we have demonstrated the algorithm to convert the nodes of the graph to the points of a metric space preserving the community structure of the graph. The algorithm depends on the sub modules 1) construction of  $L_x$  ( $L_1$  or  $L_2$ ) and 2) obtaining a structure preserving distance function. The algorithm works as picking pair of nodes from  $L_x$  and computing distance defined in the second module.

#### $L_x$ construction

The  $L_1$  is defined as  $L_1 = D(\lambda) + A$ , where  $A$  is the adjacency matrix of the given network and  $D(\lambda)$  is a diagonal matrix of same size with diagonal values equal to a non negative constant  $\lambda$ .

The  $L_2$  is defined as  $L_2 = D(\lambda_x) + A$ , where  $A$  is the adjacency matrix of the given network and  $D(\lambda_x)$  is a diagonal matrix of same size with diagonal values determine by a non negative function  $\lambda_x$  of the node  $x$ .

The choice of  $\lambda$  and  $\lambda_x$  plays a crucial role in combination with the function chosen in the second module for determination of a suitable metric and is discussed later part of this subsection.

## Function selection

The function selection module determine the metric for pair of nodes. The function selector  $\phi$  converts a similarity function (Pearson coefficient( $\sigma_{PC}$ ), Spacerman coefficient( $\sigma_{SC}$ ) or Cosine similarity( $\sigma_{CS}$ )) into a distance matrix. In general the similarity function satisfies the positivity and similarity condition of metric but not triangle inequality.  $\phi$  is a metric preserving ( $\phi(d(x_i, x_j)) = d_\phi(x_i, x_j)$ ), concave and monotonically increasing function. The three conditions above refer to as chord condition. The  $\phi$  function is chosen to have minimum internal area with chord.

## Choice of $\lambda$ and $\phi(\sigma)$

The choices in the above sub modules play a crucial role in the graph to metric transformation algorithm to be used for community detection. The complex network is characterized by small average diameter and high clustering coefficient. Several studies on network structure analysis reveal that there are hub nodes and local nodes characterizing the interesting structure of the complex network. Suppose we have taken  $\phi = \arccos$ ,  $\sigma_{CS}$  and constant  $\lambda \geq 0$ .  $\lambda = 0$  penalize the effect of direct edge in the metric and is suitable to extract communities from highly dense graph.  $\lambda = 1$  place the similar weight of direct edge and common neighbor reduce the effect of direct edge in the metric and is suitable to extract communities from moderately dense graph.  $\lambda = 2$  set more importance to direct edge than common neighbor (this is the common case of available real networks).  $\lambda \geq 2$  penalize the effect of common neighbor in the metric and is suitable to extract communities from very sparse graph.

The choice of  $\lambda$  depends on the *DCC* value (2.4.4) of the input graph, i.e. whether it is sparse or dense and its cluster structure.

The algorithm for transforming a graph to the points of a metric space is given below

1.

---

**Algorithm 1** Mapping a graph into Metric space

---

**Require:**  $G = (V, E)$

**Ensure:**  $M = (V, d)$

- 1:  $D_{(n \times n)}^\lambda = \begin{cases} 0 & \text{if } i \neq j \\ \lambda \geq 0 & \text{if } i = j \end{cases}$
  - 2:  $A = D^\lambda + E$
  - 3: **for**  $i = 1$  to  $n$  **do**
  - 4:   **for**  $j = 1$  to  $n$  **do**
  - 5:      $d(v_i, v_j) = \phi(1 - \frac{a_i \cdot a_j}{|a_i||a_j|})$ , where  $v_i, v_j \in V$  and  $a_k$  is the  $k^{th}$  row of  $A$  and  $\phi$  is an affine function.
  - 6:   **end for**
  - 7: **end for**
  - 8: **return**  $M = (V, d)$
- 

**Theorem 2.3.1**  $M = (V, d)$  constructed in the above algorithm 1 is a metric space with respect to the metric  $d$ , i.e.

The proof of the theorem is straight forward and satisfies the following metric properties.

- $d(v_i, v_j) \geq 0$  &  $d(v_i, v_i) = 0$
- $d(v_i, v_j) = d(v_j, v_i)$
- $d(v_i, v_j) \leq d(v_i, v_k) + d(v_k, v_j)$

## 2.4 Community detection on induced metric space

In this section we have explore k partitioning algorithm for the purpose of network community detection by using the metric space constructed above for each graph. We have

also studied and analyzed the advantages of the k partitioning method over the standard algorithm for network community detection.

### 2.4.1 k-partitioning

The community detection methods based on K-partitioning of graph is possible using newly defined node distance, because the nodes of the graph are converted into the points of a metric space. The k-partitioning of graph using this distance converges automatically and does not compute the value of objective function in iterations therefore reduce the computation compared to standard graph partitioning methods. The results of k-partitioning of graph using metric are competitive on a large set networks shown in section 2.5. The algorithm for community detection using k-partitioning and its details analysis is given below. Before that we need to determine the value of k and is discussed in the next subsection.

### 2.4.2 Initialization

Determining the optimal number of k is an important problem for community detection researchers. An extensive analysis can be found in the work of Leskovec et al. [45]. The set of initial nodes are also very important problem for k partitioning algorithm. The standard practice is to solve an optimization equation with respect to k for which the optimal value of the objective function is achieved. Another method based on farthest first traversal is also very useful in terms of computational efficiency [35]. For small networks the global optimization works better and for very large network the second choice give the faster approximate solution.

- Input: graph  $G = (V, E)$ , with the node similarity  $sim(x_a, x_b)$  defined on it

- Output: A partition of the nodes into  $k$  communities  $C_1, C_2, \dots, C_k$
- Objective function: Maximize the minimum intra community similarity

$$\min_{j \in \{1, 2, \dots, k\}} \max_{x_a, x_b \in C_j} \text{sim}(x_a, x_b)$$

---

**Algorithm 2** k-center partitioning algorithm

---

**Require:**  $M = (V, d)$

**Ensure:**  $T = \{C_1, C_2, \dots, C_k\}$  with minimum  $\text{cost}(T)$

```

1: Initialize centers  $z_1, \dots, z_k \in R^n$  and clusters  $T = \{C_1, C_2, \dots, C_k\}$ 
2: repeat
3:   for  $i = 1$  to  $k$  do
4:     for  $j = 1$  to  $k$  do
5:        $C_i \leftarrow \{x \in V \text{ s.t. } |z_i - x| \leq |z_j - x|\}$ 
6:     end for
7:   end for
8:   for  $j = 1$  to  $k$  do
9:      $z_i \leftarrow \text{mean}(C_i)$ 
10:  end for
11: until  $|\text{cost}(T_t) - \text{cost}(T_{t+1})| = 0$ 
12: return  $T = \{C_1, C_2, \dots, C_k\}$ 

```

---

### 2.4.3 Convergence

Convergence of the network community detection algorithms are the least studied research areas of network science. However, the rate of convergence is one of the important issues and low rate of convergence is the major pitfall of the most of the existing algorithms. Due to the transformation into the metric space, our algorithm equipped with the quick convergence facility of the k-partitioning on metric space by providing a good set of initial points. Another crucial pitfall suffer by majority of the existing algorithms is the validation of the objective function in each iteration during convergence. Our algorithm



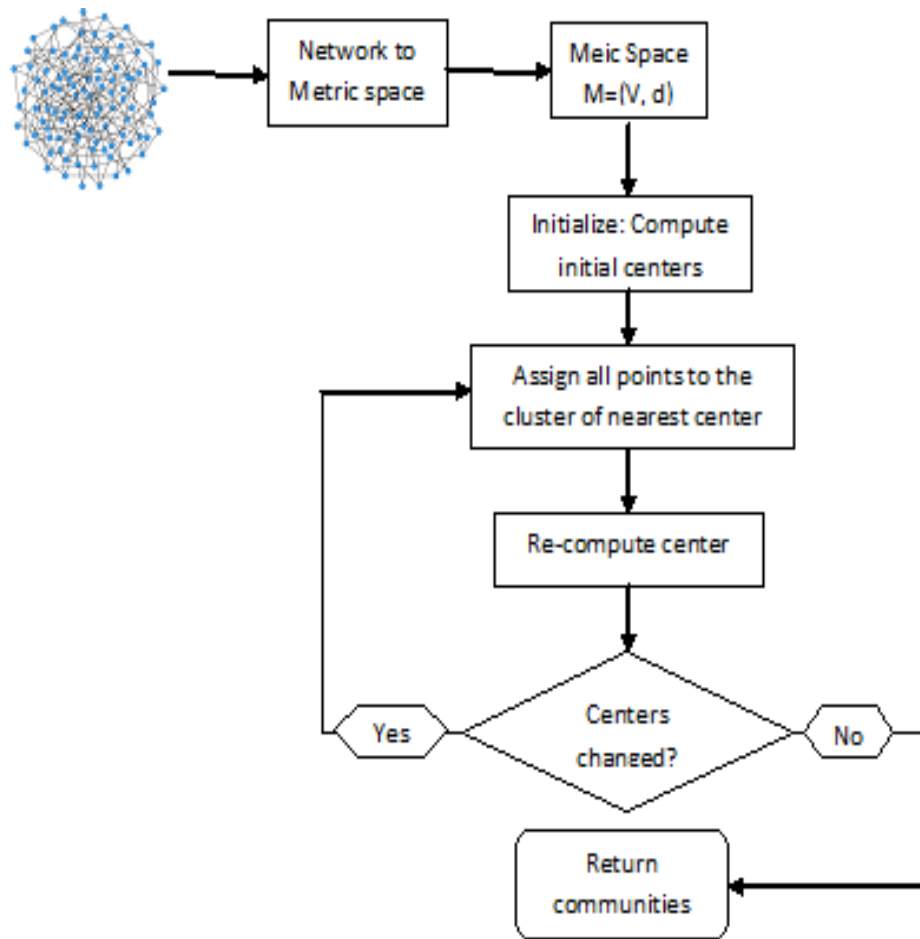


Figure 2.1: Block diagram of k-central algorithm for complex network via metric space transformation

converges automatically to the optimal partition thus reduces the cost of validation during convergence.

**Theorem 2.4.1** *During the course of the  $k$  center partitioning algorithm, the cost monotonically decreases.*

**Proof** Let  $Z^t = \{z_1^t, \dots, z_k^t\}$ ,  $T^t = \{C_1^t, \dots, C_k^t\}$  denote the centers and clusters at the start of the  $t^{th}$  iteration of  $k$  partitioning algorithm. The first step of the iteration assigns each data point to its closest center; therefore  $cost(T^{t+1}, Z^t) \leq cost(T^t, Z^t)$

On the second step, each cluster is re-centered at its mean; therefore  $cost(T^{t+1}, Z^{t+1}) \leq cost(T^{t+1}, Z^t)$

**Theorem 2.4.2** *If  $T$  is the solution returned by farthest-first traversal, and  $T^o$  is the optimal solution, then  $cost(T^o) \leq cost(T) \leq 2cost(T^o)$*

**Proof** The proof of the theorem can be obtained in the article [35].

#### 2.4.4 Data complexity

The key characteristics of complex network are "high clustering coefficient" and "small average path length". The first property justifies the community structure of the network whereas the second property justifies the small world phenomena of real networks. Given a network, that is, given number of nodes and number of edges what are the bounds of average distance and clustering coefficient? The two properties of Optimal Complex Network (OCN) are 1) minimum possible average distance and 2) maximum possible clustering coefficient. There is usually a unique graph with the largest average clustering, which at the same time has the smallest possible average distance. In contrast, there are many graphs with the same minimum average distance, ignoring their average clustering.

The objective of this work is to measure the community detectability of the complex network,  $G(N, m, L, C)$ , where  $N$  is the number of vertices,  $m$  is the number of edges,  $L$  is the average path length and  $C$  is the average clustering coefficient.

Average path length:  $L_{N,m}$  The smallest possible average distance of a graph with  $N$  vertices and  $m$  edges, which we denote  $L_{N,m} = \frac{1}{m} \sum_{u,v \in E} d(u, v)$ .

Clustering coefficient: If  $d_u (> 1)$  is the degree of a vertex  $u$ , and  $t_u$  is the number of edges among its neighbors, its clustering coefficient is,  $C(u) = t_u / \binom{d_u}{2}$

In some graph community detection is easy and most of the algorithms work very well (e.g. disjoint cliques). On the other-hand, in some graph community detection is very difficult rarely some algorithms work well (e.g. circular graph)

Data complexity of community detection: Informally, Given a graph with  $N$  vertices and  $m$  edges  $G(N, m)$ , with what extent we can reveal the community structure, is the data complexity for community detection of that graph. Data complexity for community detection (DCC) is denoted as  $(\alpha(G(N, m, L, C)))$ ,  $\alpha(G(N, m, L, C))$  near to 0 for a graph easy to detect community and  $\alpha(G(N, m, L, C))$  near to 1 no community structure. DCC is calculated as the ratio between common edges of  $G^*(N, m, L, C)$  and  $G(N, m, L, C)$  with  $m$  the number of edges of  $G$  or  $G^*$ , where  $G^*(N, m, L, C)$  is a graph with same average path length constructed by adding minimum number of edges to an empty graph of  $N$  nodes followed by addition of more edges to obtain total number  $m$  by maximizing clustering coefficient.

Higher value of DCC for a particular network signifies that we can extract good community structure of the network; however a lower value of DCC signifies that none of the algorithms are very much useful to capture community structure of the network. Other advantage of DCC is that it can assess the quality of an algorithm. When DCC is high

and value of evaluation measure is low it simply signifies that there are enough room to improve the algorithm.

## 2.5 Experiments and results

We performed many of experiments to test the proposed network detection method via induced metric space over several real networks<sup>2.2</sup>. Objective of the experiment is to verify behavior of the algorithm and the time required to compute the algorithm. One of the major goals of the experiment is to see the behavior of the algorithm with respect to the change of values of the crucial limits of the data and the parameters of the algorithm.

Experiments are also conducted to compare the results (tables 2.3, 2.4 and 2.5) of our algorithm with the state of the art algorithms (table 2.1) available in the literature in terms of common measures mostly used by the researchers of the domain of network community detection. The details of the several experiments and the analysis of the results are given in the following subsections.

### 2.5.1 Experimental designs

Experiment for comparison: In this experiment we have compared several algorithms for network community detection with the one constructed using proposed distance. Experiment is performed on a large list of network data sets. Two version of the experiment is developed for comparison purpose based on two different quality measure conductance and modularity. The results are shown in the tables 2.3 and 2.4 respectively.

Experiment on performance and time: In this experiment we have evaluated our algorithm for performance on the network collection<sup>2.2</sup>. We have evaluated the time taken by our algorithm on different size of networks and is shown in the table 2.5.

### 2.5.2 Performance indicator

**Modularity:** The notion of modularity is the most popular for the network community detection purpose. The modularity index assigns high scores to communities whose internal edges are more than that expected in a random-network model which preserves the degree distribution of the given network.

**Conductance:** Conductance is widely used for graph partitioning literature. The conductance of a set  $S$  with complement  $S^C$  is the ratio of the number of edges connecting nodes in  $S$  to nodes in  $S^C$  by the total number of edges incident to  $S$  or to  $S^C$  (whichever number is smaller).

### 2.5.3 Datasets

A list of real networks taken from several real life interactions is considered for our experiments and they are tabulate 2.2 below. We have also listed the number of nodes, number of edges, average diameter, data complexity for community detection (DCC) and the  $k$  value used (2.4.2). The values of the last column can be used to assess the quality of detected communities as discussed in the subsection 2.4.4.

### 2.5.4 Computational results

In this subsection we have compared two groups of algorithms for network community detection with our proposed algorithm based on metric space. Experiment is performed on a large list of network data sets. Two version of the experiment is developed for comparison purpose based on two different quality measure conductance and modularity. The results based on conductance is shown in the table 2.3 and the results based on modularity is shown in the table 2.4, respectively. Regarding the two groups of algorithms; first group

Table 2.2: Complex network datasets and values of their parameters

Name	Type	# Nodes	# Edges	Diameter	DCC	k
Facebook	U	4,039	88,234	4.7	0.72498	164
Gplus	D	107,614	13,673,453	3	0.50073	457
Twitter	D	81,306	1,768,149	4.5	0.57072	213
Epinions1	D	75,879	508,837	5	0.14001	128
LiveJournal1	D	4,847,571	68,993,773	6.5	0.27432	117
Pokec	D	1,632,803	30,622,564	5.2	0.10971	246
Slashdot0811	D	77,360	905,468	4.7	0.05884	81
Slashdot0922	D	82,168	948,464	4.7	0.06340	87
Friendster	U	65,608,366	1,806,067,135	5.8	0.16231	833
Orkut	U	3,072,441	117,185,083	4.8	0.16689	756
Youtube	U	1,134,890	2,987,624	6.5	0.08090	811
DBLP	U	317,080	1,049,866	8	0.63307	268
Arxiv-AstroPh	U	18,772	396,160	5	0.65841	23
web-Stanford	D	281,903	2,312,497	9.7	0.60034	69
Amazon0601	D	403,394	3,387,388	7.6	0.41890	92
P2P-Gnutella31	D	62,586	147,892	6.5	0.00710	35
RoadNet-CA	U	1,965,206	5,533,214	500	0.40458	322
Wiki-Vote	D	7,115	103,689	3.8	0.17048	21

contain algorithms based on semi-definite programming and the second group contain algorithms based on graph traversal approaches. For each group, we have taken the best value of conductance in table 2.3 and best value of modularity in table 2.4 among all the algorithms in the groups. The results obtained with our approach are very competitive with most of the well known algorithms in the literature and this is justified over the large collection of datasets. On the other hand, it can be observed that time taken (table 2.5) by our algorithm is quite less compared to other methods and justify the theoretical findings described in sections 2.3 and 3.5.2.

Table 2.3: Comparison of our approaches with other best methods in terms of conductance, the number inside the brackets denotes the algorithm of the group

Name	Spectral	SDP	GT	Metric
Facebook	0.0097(5)	0.1074(3)	0.1044(7)	0.1082
Gplus	0.0119(5)	0.1593(3)	0.1544(7)	0.1602
Twitter	0.0035(5)	0.0480(3)	0.0465(7)	0.0483
Epinions1	0.0087(5)	0.1247(6)	0.1208(7)	0.1254
LiveJournal1	0.0039(5)	0.0703(6)	0.0680(7)	0.0706
Pokec	0.0009(4)	0.0174(3)	0.0168(7)	0.0175
Slashdot0811	0.0005(5)	0.0097(6)	0.0094(7)	0.0098
Slashdot0922	0.0007(4)	0.0138(3)	0.0133(5)	0.0138
Friendster	0.0012(5)	0.0273(1)	0.0263(7)	0.0273
Orkut	0.0016(5)	0.0411(3)	0.0397(7)	0.0412
Youtube	0.0031(5)	0.0869(3)	0.0838(7)	0.0871
DBLP	0.0007(4)	0.0210(3)	0.0203(7)	0.0211
Arxiv-AstroPh	0.0024(5)	0.0929(6)	0.0895(7)	0.0931
web-Stanford	0.0007(5)	0.0320(1)	0.0308(7)	0.0320
Amazon0601	0.0018(5)	0.0899(6)	0.0865(7)	0.0900
P2P-Gnutella31	0.0009(5)	0.0522(6)	0.0503(7)	0.0523
RoadNet-CA	0.0024(5)	0.1502(3)	0.1445(7)	0.1504
Wiki-Vote	0.0026(5)	0.1853(6)	0.1783(7)	0.1855

Table 2.4: Comparison of our approaches with other best methods in terms of modularity, the number inside the brackets denotes the algorithm of the group

Name	Spectral	SDP	GT	Metric
Facebook	0.4487(1)	0.5464(4)	0.5434(5)	0.5472
Gplus	0.2573(1)	0.4047(3)	0.3998(5)	0.4056
Twitter	0.3261(3)	0.3706(1)	0.3691(7)	0.3709
Epinions1	0.0280(1)	0.1440(3)	0.1401(5)	0.1447
LiveJournal1	0.0791(1)	0.1455(5)	0.1432(5)	0.1458
Pokec	0.0129(3)	0.0294(1)	0.0288(5)	0.0295
Slashdot0811	0.0038(1)	0.0130(4)	0.0127(7)	0.0131
Slashdot0922	0.0045(1)	0.0176(5)	0.0171(5)	0.0176
Friendster	0.0275(4)	0.0536(5)	0.0526(7)	0.0536
Orkut	0.0294(3)	0.0689(4)	0.0675(5)	0.0690
Youtube	0.0096(1)	0.0934(2)	0.0903(5)	0.0936
DBLP	0.4011(5)	0.4214(1)	0.4207(5)	0.4215
Arxiv-AstroPh	0.4174(3)	0.5079(3)	0.5045(5)	0.5081
web-Stanford	0.3595(5)	0.3908(4)	0.3896(7)	0.3908
Amazon0601	0.1768(1)	0.2649(4)	0.2615(7)	0.2650
P2P-Gnutella31	0.0009(1)	0.0522(2)	0.0503(5)	0.0523
RoadNet-CA	0.0212(3)	0.1690(4)	0.1633(5)	0.1692
Wiki-Vote	0.0266(1)	0.2093(1)	0.2023(5)	0.2095

Table 2.5: Comparison of our approaches with other best methods in terms of time(second)

Algorithm	Spectral	SDP	GT	Metric
Minimum Time	884	910	871	869
Maximum Time	1386	1725	1641	869
Average Time	917	981	1338	869



### 2.5.5 Parameter settings

The values of the several parameters are very crucial in our algorithm. Here we have discussed the different settings of  $k$ ,  $\lambda$ , DCC and affine function. For each data described in table 2.2 the  $k$  value is obtained by optimizing the conductance value as described in sub-section 2.4.2 and the values are provided in table 2.2. For small data sets (not considered for our experiments) the results are very sensitive to  $k$  whereas, for large networks (all of above list) the results are less sensitive to  $k$ . The value  $\lambda$  is taken  $\lambda = 2$  in all the computation above, however the results can be improved more by optimizing  $\lambda$ . The DCC value provide us prior information about the community structure, it can be observed that we obtained good community structure where DCC value is high. In all the experiments described above the  $\phi(\sigma)()$  is constructed with arccos function and cosine similarity.

### 2.5.6 Results analysis and achievements

In this subsection, we have described the analysis of the results obtained in our experiments shown above and also highlighted the achievements from the results. It is clearly evident from the results shown in the tables 2.3, 2.4 and 2.5 that, proposed metric based method for network community detection provide very good competitive performance with respect to conductance modularity and time. However, a good community detection algorithm must provide the results close to the unknown optimal community structure. To assess the optimality we have considered the best results of each class of algorithms and treated them as one of the best known estimate to the optimal community structure of the network. It is also evident from the results that our method provide results very close to considered estimates of optimal communities.

## 2.6 Conclusions

Network community detection became an important research problem in recent years. In this chapter we have demonstrated and analyzed a new approaches to network community detection via metric space induced by the graph. The main achievement of the work was to use the rich literature of clustering in metric space. Clustering is easy NP-Hard in metric space, whereas network community detection is NP-Hard. The results obtained with our approach were very competitive with most of the well known algorithms in the literature and justified over the large collection of datasets. Our algorithm converges automatically to optimal clustering. One important advantage of the proposed method is no need to verify objective function value to guide next iteration, like popular approaches, thus saving the time of computation.

## Chapter 3

# Nearest Neighbor search in the Metric Space of Complex Network for Community Detection

The objective of this chapter is to bridge the gap between two important research directions, 1) nearest neighbor search, which is a fundamental computational tool for large data analysis, and 2) complex network analysis, which deals with large real graphs but generally studied via graph theoretic analysis or spectral analysis. In this chapter, we have studied the nearest neighbor search problem in complex network by the development of a suitable notion of nearness. The computation of efficient nearest neighbor search among the nodes of a complex network using the metric tree and locality sensitive hashing (LSH) are also studied and experimented. For evaluation of the proposed nearest neighbor search on complex network, we applied it in network community detection problem. Experiments are performed to verify the usefulness of nearness measures for the complex networks, the role of metric tree and LSH to compute fast and approximate node nearness and

the efficiency of community detection using nearest neighbor search. We observed that nearest neighbor between network nodes is a very efficient tool to explore better community structure of the real networks, several efficient approximation scheme are very useful for large networks, which hardly made any degradation of results, whereas saves lot of computational times and nearest neighbor based community detection approach is very competitive in terms of efficiency and time.

### 3.1 Introduction

The nearest neighbor (NN) search is an important computational primitive for structural analysis of data and other query retrieval purposes. NN search is very useful for dealing with massive data sets, but it suffers with curse of dimension [72, 75]. Though nearest neighbor search is a extensively studied research problem for low dimensional data, a recent surge of results shows that it is most useful tool for analyzing very large data, provided a suitable space partitioning data structure is used, like, kd-tree, quad-tree, R-tree, metric-tree and locality sensitive hashing [20, 32, 38, 59]. One more advantage of using nearest neighbor search for large data analysis is the availability of efficient approximation scheme, which provides almost same results in very less time [3, 48]. Though nearest neighbor search is very successful and extensively used across the research domains of computer science, it is not studied rigorously in complex network analysis. Complex networks are generally studied in graph theoretic framework or in spectral analysis framework. One basic reason for this limitation may be the nodes of the complex networks are not naturally lie on a metric space, thus restricting the use of nearest neighbor analysis which is done using metric or nearness measures.

Other than graphs, the complex networks are characterized by small "average path

length” and high ”clustering coefficient”. A network community (also known as a module or cluster) is typically a group of nodes with more interconnections among its members than the remaining part of the network [29, 74, 78]. To extract such group of nodes from a network one generally selects an objective function that captures the possible communities as a set of nodes with better internal connectivity than external [49, 54]. However, very less research is done for network community detection, which tries to develop nearness between the nodes of a complex network and use the nearest neighbor search for partitioning the network [8, 9, 12, 26, 47, 50, 65]. The way metric is defined among the nodes should be able to capture the crucial properties of complex networks. Therefore, we need to create the metric very carefully so that it can explore the underlying community structure of the real life networks [73].

Extracting network communities in large real graphs such as social networks, web, collaboration networks and bio-networks is an important research direction of recent interest [11, 29, 31, 56, 56, 66]. In this work, we have developed the notion of nearness among the nodes of the network using some new matrices derived from the modified adjacency matrix of the graph which is flexible over the networks and can be tuned to enhance the structural properties of the network required for community detection.

The main contributions of this work are, 1) development of the concept of nearness between the nodes of a complex network, 2) comparing the proposed nearness with other notions of similarities, 3) study and experiment on approximate nearest neighbor search for complex network using M-tree and LSH, 4) design of efficient community detection algorithm using nearest neighbor search. We observed that nearest neighbor between network nodes is a very efficient tool to explore better community structure of the real networks. Further several efficient approximation scheme are very useful for large networks, which hardly made any degradation of results, whereas saves lot of computational

times.

The rest of this chapter is organized as follows. Section 3.2 describes the notion of nearness in complex network and proposed method to compute distance between the nodes of a complex network. Section 3.3 and 3.4 describe the algorithm of the nearest neighbor search over complex network using of metric tree and locality sensitive hashing methods respectively. In Section 3.5, the proposed algorithm for network community detection using nearest neighbor search is discussed. The results of the comparison between community detection algorithms are illustrated in Section 3.6.

## 3.2 Notion of nearness in complex network

The notion of nearness between the nodes of a graph, are used in several purposes in the history of literature of graph theory. Most of the time the shortest path and edge connectivity are popular choice to describe nearness of nodes. However, the edge count does not give the true measure of network connectivity. A true measure of nearness in complex network should able to determine how much one node can affect the other node to provide a better measure of connectivity between nodes of a real life complex network. Research in this direction need special attention in the domain of complex network analysis, one such is proposed in this chapter and described in the following subsections.

### 3.2.1 Definitions

**Metric space of network** Given, a graph  $G = (V, E)$  the metric is defined over the vertex set  $V$  and  $d$ , a function to compute the distance between two vertices of  $V$ . Pair  $(V, d)$  distinguished metric space if  $d$  satisfies reflexivity, non-negativity, symmetry and

triangle inequality.

**Nearest neighbor search on network** The nearest-neighbor searching problem in complex network is to find the nearest node in a graph  $G = (V, E)$  between a query vertex  $v_q$  and any other vertex of the graph  $V \setminus \{v_q\}$ , with respect to a metric space  $M(V, d)$  associated with the graph  $G = (V, E)$ .

**Approximate nearest neighbor search on network** For any  $v_q \in V$ , An  $\epsilon$  approximate NN of  $v_q \in V$  is to find a point  $v_p \in V \setminus \{v_q\}$  s.t.  $d(v_p, v_q) \leq (1 + \epsilon)d(v, v_q) \forall v \in V \setminus \{v_q\}$ .

### 3.2.2 Nearness in complex network

Methods based on node neighborhoods. For a node  $x$ , let  $N(x)$  denote the set of neighbors of  $x$  in a graph  $G(V, E)$ . A number of approaches are based on the idea that two nodes  $x$  and  $y$  are more likely to be affected by one another if their sets of neighbors  $N(x)$  and  $N(y)$  have large overlap.

**Common neighbors:** The most direct implementation of this idea for nearness computation is to define  $d(x, y) := |N(x) \cap N(y)|$ , the number of neighbors that  $x$  and  $y$  have in common.

**Jaccard coefficient:** The Jaccard coefficient, a commonly used similarity metric, measure the probability that both  $x$  and  $y$  have a feature  $f$ , for a randomly selected feature  $f$  that either  $x$  or  $y$  has. If we take features here to be neighbors in  $G(V, E)$ , this leads to the measure  $d(x, y) := |N(x) \cap N(y)| / |N(x) \cup N(y)|$ .

**Preferential attachment:** The probability that a new edge involves node  $x$  is proportional to  $|N(x)|$ , the current number of neighbors of  $x$ . The probability of co-authorship of  $x$  and  $y$  is correlated with the product of the number of collaborators of  $x$  and  $y$ . This

corresponds to the measure  $d(x, y) := |N(x)| \times |N(y)|$ .

**Katz measure:** This measure directly sums over the collection of paths, exponentially damped by length to count short paths more heavily. This leads to the measure  $d(x, y) := \beta \times |\text{paths}(x, y)|$  where,  $\text{paths}(x, y)$  is the set of all length paths from  $x$  to  $y$ . ( $\beta$  determines the path size, since paths of length three or more contribute very little to the summation.)

**Commute time:** A random walk on  $G$  starts at a node  $x$ , and iteratively moves to a neighbor of  $x$  chosen uniformly at random. The hitting time  $H(x, y)$  from  $x$  to  $y$  is the expected number of steps required for a random walk starting at  $x$  to reach  $y$ . Since the hitting time is not in general symmetric, it is also natural to consider a commute time  $C(x, y) := H(x, y) + H(y, x)$ .

**PageRank:** Random resets form the basis of the PageRank measure for Web pages, and we can adapt it for link prediction as follows: Define  $d(x, y)$  to be the stationary probability of  $y$  in a random walk that returns to  $x$  with probability  $\alpha$  each step, moving to a random neighbor with probability  $1 - \alpha$ .

Most of the methods are developed for different types of problems like information retrieval, ranking, prediction e.t.c. and developed for general graphs.

### 3.2.3 Nearness in complex network:

In this subsection, we developed the notion of nearness among the nodes of the network using some linear combination of adjacency matrix  $A$  and identity matrix of same dimension for the network  $G = (V, E)$ . The similarities between the nodes are defined on matrix  $L = \lambda I + A$  as spherical similarity among the rows and determine by applying a concave function  $\phi$  over the standard notions of similarities like, Pearson coefficient( $\sigma_{PC}$ ), Spac-



erman coefficient( $\sigma_{SC}$ ) or Cosine similarity( $\sigma_{CS}$ ).  $\phi(\sigma)()$  must be chosen using the chord condition, i.e., metric-preserving ( $\phi(d(x_i, x_j)) = d_\phi(x_i, x_j)$ ), concave and monotonically-increasing, to obtain a metric. It works by picking a pair of rows from  $L$  and computing the distance defined in the  $\phi(\sigma)()$ . The function  $\phi$  converts a similarity function (Pearson coefficient ( $\sigma_{PC}$ ), Spacerman coefficient ( $\sigma_{SC}$ ) or cosine similarity ( $\sigma_{CS}$ )) into a distance matrix. In general, the similarity function satisfies the positivity and similarity condition of the metric, but not the triangle inequality.  $\phi$  is a metric-preserving ( $\phi(d(x_i, x_j)) = d_\phi(x_i, x_j)$ ), concave and monotonically-increasing function. The three conditions above are referred to as the chord condition. The  $\phi$  function is chosen to have minimum internal area with the chord.

The choice of  $\lambda$  and  $\phi(\sigma)()$  in the above sub-modules play a crucial role in the graph to metric transformation algorithm to be used for community detection. The complex network is characterized by a small average diameter and a high clustering coefficient. Several studies on network structure analysis reveal that there are hub nodes and local nodes characterizing the interesting structure of the complex network. Suppose we have taken  $\phi = \arccos$ ,  $\sigma_{CS}$  and constant  $\lambda \geq 0$ .  $\lambda = 0$  penalizes the effect of the direct edge in the metric and is suitable to extract communities from a highly dense graph.  $\lambda = 1$  places a similar weight of the direct edge, and the common neighbor reduces the effect of the direct edge in the metric and is suitable to extract communities from a moderately dense graph.  $\lambda = 2$  sets more importance for the direct edge than the common neighbor (this is the common case of available real networks).  $\lambda \geq 2$  penalizes the effect of the common neighbor in the metric and is suitable for extracting communities from a very sparse graph.

### 3.3 Nearest neighbor search on complex network using metric tree

There are numerous methods developed to compute the nearest neighbor search for points of a metric space. However, finding the nearest neighbor search on some data where dimension is high suffer from curse of dimension. Some recent research on this direction revealed that dimension constrained can be tackled by using efficient data structures like metric tree and locality sensitive hashing. In this section we have explored metric tree to perform the nearest neighbor search on complex network with the help of metric mapping of complex network described in the previous section.

#### 3.3.1 Metric-tree

A metric tree is a data structure specially designed to perform the nearest neighbor query for the points residing on a metric space and perform well on high dimension particularly when some approximation is permitted. A metric tree organizes a set of points in a spatial hierarchical manner. It is a binary tree whose nodes represent a set of points. The root node represents all points, and the points represented by an internal node  $v$  is partitioned into two subsets, represented by its two children. Formally, if we use  $N(v)$  to denote the set of points represented by node  $v$ , and use  $v.lc$  and  $v.rc$  to denote the left child and the right child of node  $v$ , then we have  $N(v) = N(v.lc) \cup N(v.rc)$   $\phi = N(v.lc) \cap N(v.rc)$  for all the non-leaf nodes. At the lowest level, each leaf node contains very few points.

An M-Tree [15] consists of leaf node, internal node and routing object. Leaf nodes are set of objects  $N_v$  with pointer to parent object  $v_p$ . Internal nodes are set of routing objects  $N_{RO}$  with pointer to its parent object  $v_p$ . Routing object  $v_r$  store covering radius

$r(v_r)$  and pointer to covering tree  $T(v_r)$ , Distance of  $v_r$  from its parent object  $d(v_r, P(v_r))$ . Feature values stored in the object  $v_j$  are object identifier oid ( $v_j$ ) and distance of  $v_j$  from its parent object  $d(v_j, P(v_j))$

The key to building a metric-tree is how to partition a node  $v$ . A typical way is as follows: We first choose two pivot points from  $N(v)$ , denoted as  $v.lpv$  and  $v.rpv$ . Ideally,  $v.lpv$  and  $v.rpv$  are chosen so that the distance between them is the largest of all distances within  $N(v)$ . More specifically,  $\|v.lpv - v.rpv\| = \max_{p_1, p_2 \in N(v)} \|p_1 - p_2\|$ . A search on a metric-tree is performed using a stack. The current radius  $r$  is used to decide which child node to search first. If the query  $q$  is on the left of current point, then  $v.lc$  is searched first, otherwise,  $v.rc$  is searched first. At all times, the algorithm maintains a candidate NN and there distance determines the current radius, which is the nearest neighbor it finds so far while traversing the tree. The algorithm for nearest neighbor search using metric tree is given below 3.

### 3.3.2 Nearest Neighbor search algorithm using M-Tree

algorithm

---

**Algorithm 3** NN search in M-Tree

---

**Require:**  $M = (V, d)$  &  $q$

**Ensure:**  $d(q, v_q)$

- 1: Insert root object  $v_r$  in stack
  - 2: Set current radius as  $d(v_r, q)$
  - 3: Successively traverse the tree in search of  $q$
  - 4: PUSH all the objects of traversal path into stack
  - 5: Update the current radius
  - 6: If leaf object reached
  - 7: POP objects from stack
  - 8: For all points lying inside the ball of current radius centering  $q$ , verify for possible nearest neighbor and update the current radius.
  - 9: **return**  $d(q, v_q)$
-

The theoretical advantage of using metric tree as a data structure for nearest neighbor search is: Let  $M = (V, d)$ , be a bounded metric space. Then for any fixed data  $V \in R^n$  of size  $n$ , and for constant  $c \geq 1$ ,  $\exists \epsilon$  such that we may compute  $d(q, V)|_\epsilon$  with at most  $c \cdot \lceil \log(n) + 1 \rceil$  expected metric evaluations [16].

### 3.4 Nearest neighbor search on complex network using locality sensitive hashing

Metric trees, so far represent the practical state of the art for achieving efficiency in the largest dimension possible. However, many real-world problems consist of very large dimension and beyond the capability of such search structures to achieve sub-linear efficiency. Thus, the high-dimensional case is the long-standing frontier of the nearest-neighbor problem. The approximate nearest neighbor can be computed very efficiently using Locality sensitive hashing.

#### 3.4.1 Approximate nearest neighbor

Given a metric space  $(S, d)$  and some finite subset  $S_D$  of data points  $S_D \subset S$  on which the nearest neighbor queries are to be made, our aim to organize  $S_D$  s.t. NN queries can be answered more efficiently. For any  $q \in S$ , NN problem consists of finding single minimal located point  $p \in S_D$  s.t.  $d(p, q)$  is minimum over all  $p \in S_D$ . We denote this by  $p = NN(q, S_D)$ .

An  $\epsilon$  approximate NN of  $q \in S$  is to find a point  $p \in S_D$  s.t.  $d(p, q) \leq (1 + \epsilon)d(x, d) \forall x \in S_D$ .

### 3.4.2 Locality Sensitive Hashing (LSH)

Several methods to compute first nearest neighbor query exists in the literature and locality-sensitive hashing (LSH) is most popular because of its dimension independent run time [51, 60]. In a locality sensitive hashing, the hash function has the property that close points are hash into same bucket with high probability and distance points are hash into same bucket with low probability. Mathematically, a family  $H = \{h : S \rightarrow U\}$  is called  $(r_1, r_2, p_1, p_2)$ -sensitive if for any  $p, q \in S$

- if  $p \in B(q, r_1)$  then  $Pr_H[h(q) = h(p)] \geq p_1$
- if  $p \notin B(q, r_2)$  then  $Pr_H[h(q) = h(p)] \leq p_2$

where  $B(q, r)$  denotes a hyper sphere of radius  $r$  centered at  $q$ . In order for a locality-sensitive family to be useful, it has to satisfy inequalities  $p_1 > p_2$  and  $r_1 < r_2$  when  $D$  is a distance, or  $p_1 > p_2$  and  $r_1 > r_2$  when  $D$  is a similarity measure [33, 39]. The value of  $\delta = \log(1/P_1)/\log(1/P_2)$  determines search performance of LSH. Defining a LSH as  $a(r, r(1 + \epsilon), p_1, p_2)$ , the  $(1 + \epsilon)$  NN problem can be solved via series of hashing and searching within the buckets [21, 32, 41].

### 3.4.3 Locality sensitive hash function for complex network

In this sub-section, we discuss the existence of locality sensitive hash function families for the proposed metric for complex network. The LSH data structure stores all nodes in hash tables and searches for nearest neighbor via retrieval. The hash table is contain many buckets and identified by bucket id. Unlike conventional hashing, the LSH approach tries to maximize the probability of collision of near items and put them into same bucket. For any given the query  $q$  the bucket  $h(q)$  considered to search the nearest node. In general

$k$  hash functions are chosen independently and uniformly at random from hash family  $H$ . The output of the nearest neighbor query is provided from the union of  $k$  buckets. The consensus of  $k$  functions reduces the error of approximation. For metric defined in the previous section 3.2 we considered  $k$  random points from the metric space. Each random point  $r_i$  define a hash function  $h_i(x) = \text{sign}(d(x, r_i))$ , where  $d$  is the metric and  $i \in [1, k]$ . These randomized hash functions are locality sensitive [5, 13].

---

**Algorithm 4** NN search in LSH

---

**Require:**  $M = (V, d)$  &  $q$

**Ensure:**  $d(q, V)$

- 1: Identify buckets of query point  $q$  corresponding to different hash functions.
  - 2: Compute nearest neighbor of  $q$  only for the points inside the selected buckets.
  - 3: **return**  $d(q, V)$
- 

The theoretical advantage of using locality sensitive hashing as a data structure for nearest neighbor search is: Let  $M = (V, d)$ , be a bounded metric space. Then for any fixed data  $V \in R^n$  of size  $n$ , and for constant  $c \geq 1$ ,  $\exists \epsilon$  such that we may compute  $d(q, V)|_\epsilon$  with at most  $mn^{O(1/\epsilon)}$  expected metric evaluations, where  $m$  is the number of dimension of the metric space. In case of complex network  $m = n$  so expected time is  $n^{O(2/\epsilon)}$  [16, 37].

## 3.5 Proposed community detection based on nearest neighbor

In this section we have described the algorithm proposed for network community detection using nearest neighbor search. Our approach differs from the existing methods of community detection. The broad categorization of the available algorithms is generally based on graph traversal, semidefinite programming and spectral analysis. The basic ap-

proach and the complexity of very popular algorithms are already listed in the previous chapter.

### 3.5.1 Distance based community detection

There exist no algorithms in the literature of network community detection which compute direct nearest neighbor between nodes to the best of our knowledge, however concepts of nearness used in some of the algorithms and they are described below.

Walktrap Algorithm(WT): This algorithm by Pons and Latapy [65] uses a hierarchical agglomerative method. Here, the distance between two nodes is defined in terms of random walk process. The basic idea is that if two nodes are in the same community, the probability to get to a third node located in the same community through a random walk should not be very different. The distance is constructed by summing these differences over all nodes, with a correction for degree. The complexity of the algorithm is  $O(n^3)$  as reported in Latapy & Pons (walktrap, 2004 [65]).

Label Propagation Algorithm(LP): This algorithm by Raghavan et al. [67] uses the concept of node neighborhood and the diffusion of information in the network to identify communities. Initially, each node is labeled with a unique value. Then an iterative process takes place, where each node takes the label which is the most spread in its neighborhood. This process goes on until the conditions, no label change, is met. The resulting communities are defined by the last label values with the complexity  $O(n + m)$  for each iteration as reported in Raghavan et al. (label propagation, 2007 [67]).

Geometric Brownian motion(GBM): This concept was borrowed from statistical physics by Zhou et al. [82]. This method develops the notion of Brownian motion on networks to compute the influences between the nodes, which used to discover communi-

ties of social networks. The complexity of the algorithm is  $O(n^3)$ .

### **3.5.2 Proposed algorithm for network community detection using nearest neighbor search**

In this subsection we have described k-central algorithm for the purpose of network community detection by using the nearest neighbor search on complex network. The community detection methods based on partitioning of graph is possible using nearest neighbor search, because the nodes of the graph are converted into the points of a metric space. This algorithm for network community detection converges automatically and does not compute the value of objective function in iterations therefore reduce the computation compared to standard methods. The k-central algorithm for community detection is given below. Block diagram of the algorithm is given in Fig. 3.1.

### **3.5.3 Complexity and convergence**

Complexity of the network community detection algorithms are the less studied research topic of network science. However, the rate of convergence is one of the important issues of algorithmic complexity and low rate of convergence is the major pitfall of the most of the existing algorithms. Due to the transformation into the metric space, our algorithm equipped with the quick convergence facility of the k-partitioning on metric space by providing a good set of initial points. Another crucial pitfall suffer by majority of the existing algorithms is the validation of the objective function in each iteration during convergence. Our algorithm converges automatically to the optimal partition thus reduces the cost of validation during convergence.



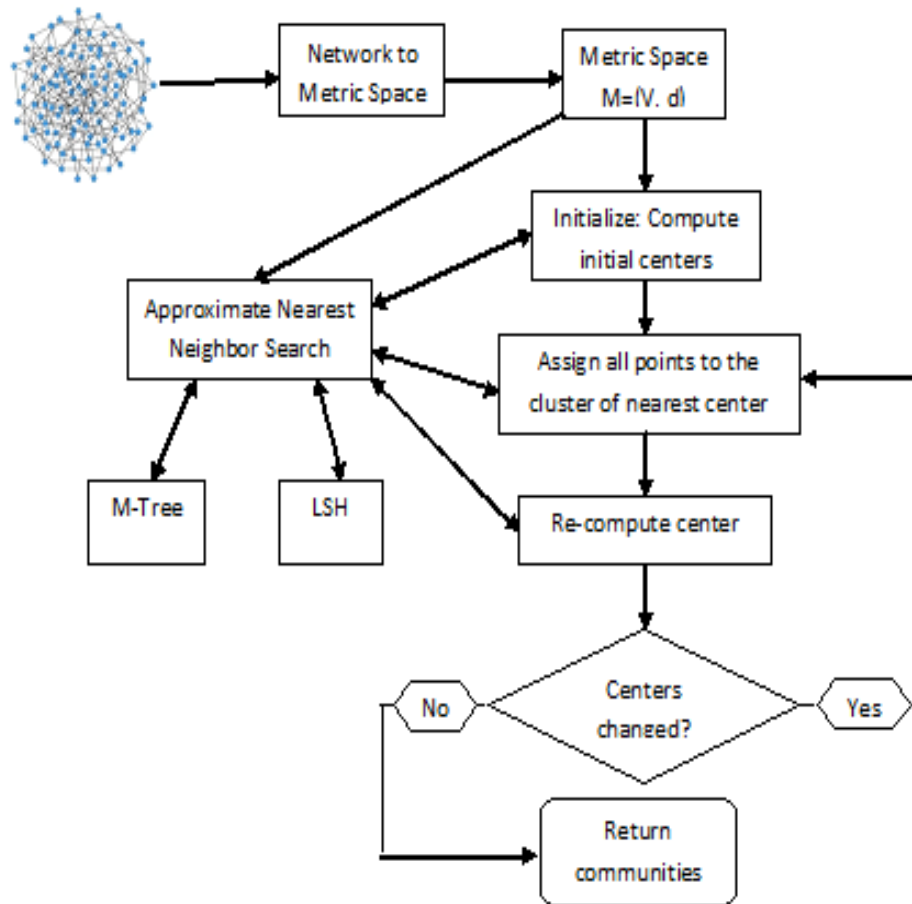


Figure 3.1: Block diagram of k-central algorithm using approximate nearest neighbor search with metric tree or locality sensitive hashing

## 3.6 Experiments and results

In this section we described in details several experiments to asses the, proposed nearness measure for the nodes of the network, efficiency of several approximation scheme to compute node nearness and performance of proposed algorithm for community detection. Several experiments conducted in this regards are detailed below along with their parameter settings, results and conclusions.

### 3.6.1 Experimental Designs

We performed three different experiments to asses the performance of the proposed network nearest neighbor search for community detection. First experiment is designed to evaluate the nearness measure, second experiment is designed to explore the effectiveness of approximate nearest neighbor search for network community detection and the third experiment is designed to verify behavior of the algorithm and the time required to compute the algorithm. One of the major goals of the last experiment is to verify the behavior of the algorithm with respect to the performance of other popular methods exists in the literature in terms of standard modularity measures. Experiments are conducted over several real networks<sup>2.2</sup> to compare the results (tables 3.3 and 3.4) of our algorithm with the state-of-the-art algorithms (table 2.1) available in the literature in terms of modularity most preferred by the researchers of the domain of network community detection. The details of the several experiments and the analysis of the results are given in the following subsections.

### 3.6.2 Datasets

A list of real networks taken from several real life interactions is considered for our experiments and they are shown in Table 2.2 (previous chapter). We have also listed the number of nodes, number of edges, average diameter, and the  $k$  value used (3.5.2). The values of the last column can be used to assess the quality of detected communities.

### 3.6.3 Exp1: Experiment with nearness measure

In this experiment we tried to assess the usefulness of proposed nearness measure between the nodes of complex network. For this purpose we have equipped our algorithm with different measures of nearness along with our measure. Experimental steps are as follows:

**Nearness measures:** Six different measures are taken for construction the distance based community detection. They are jaccard coefficient(JA), preferential attachment(PA), Katz measure(KM), commute time(CT), page rank(PR) and proposed metric(PM). details of the measures are already discussed in sub-sec. 3.2.2 and proposed metric is detailed in sub-sec. 3.2.3.

**Algorithm:** The community detection algorithm proposed in sec. 3.5 is used and exact nearest neighbor between nodes are considered and computed communities based on those different nearness measures.

**Network data:** Different types of real network data is taken, small, large, very sparse and relatively dense and they are discussed in sub-sec. 2.2.

**Results:** Compared the community structure obtained by algorithms, equipped with different measures of node nearness, in terms of modularity and shown in table 3.1.

**Observation:** It can be observed from table 3.1 that algorithm based on proposed metric (shown in column PA) provides better modularity than other for community detection.

Table 3.1: Exp1: Experiment with nearness measure

Name	JC	PA	KM	CT	PR	PM
Facebook	0.4806	0.4937	0.5037	0.4973	0.5206	0.5434
Gplus	0.3061	0.3253	0.3411	0.3309	0.3671	0.3998
Twitter	0.3404	0.3465	0.3508	0.3481	0.3582	0.3691
Epinions1	0.0667	0.0816	0.0943	0.0861	0.1150	0.1401
LiveJournal1	0.1010	0.1097	0.1167	0.1122	0.1284	0.1432
Pokec	0.0183	0.0205	0.0222	0.0211	0.0251	0.0288
Slashdot0811	0.0066	0.0080	0.0087	0.0082	0.0101	0.0127
Slashdot0922	0.0086	0.0105	0.0116	0.0109	0.0137	0.0171
Friendster	0.0360	0.0395	0.0422	0.0405	0.0467	0.0526
Orkut	0.0424	0.0476	0.0518	0.0491	0.0587	0.0675
Youtube	0.0375	0.0483	0.0574	0.0515	0.0724	0.0903
DBLP	0.4072	0.4103	0.4118	0.4110	0.4148	0.4207
Arxiv-AstroPh	0.4469	0.4590	0.4682	0.4624	0.4837	0.5045
web-Stanford	0.3693	0.3738	0.3765	0.3749	0.3815	0.3896
Amazon0601	0.2057	0.2174	0.2266	0.2207	0.2419	0.2615
P2P-Gnutella31	0.0180	0.0246	0.0302	0.0266	0.0394	0.0503
RoadNet-CA	0.0701	0.0893	0.1051	0.0949	0.1312	0.1633
Wiki-Vote	0.0874	0.1109	0.1308	0.1179	0.1633	0.2023

### 3.6.4 Exp2: Experiment on approximation

In this experiment we explore the effectiveness of several approximation techniques of nearest neighbor search on complex network designed via metric tree and locality sensitive hashing. For this purpose we have equipped our algorithm with different data structures (metric tree and LSH) with varying approximation ratio. Experimental steps are as follows:

**Metric and algorithm:** The algorithms considered in this experiment used proposed measures of node nearness detailed in sub-sec. 3.2.3. The community detection algorithm proposed in sec. 3.5 is used in this experiment.

**Approximation:** Computed communities using approximate nearest neighbor via metric tree and locality sensitive hashing. Different precision of approximation is considered ranges from 0 – 0.5 and computed five times each over both the scheme of approximation.

**Network data:** Different types of real network data is taken to verify the acceptability of degradation over the networks and is shown in table 3.2.

**Results:** Compared the community structure obtained by algorithms, equipped with approximate nearest neighbor instead of exact measures of node nearness, in terms of modularity and shown in table 3.2.

**Observations:** Observed that both the approximation schemes are very good for community detection and slightly degrade the results under ranges of approximations. 3.2

### 3.6.5 Exp3: Experiment to evaluate proposed algorithm

In this experiment we have compared several algorithms for network community detection with our proposed algorithm developed using the nearest neighbor search in complex network, which is discussed in sec.3.5. Experiment is performed on a large list of network data sets 2.2. Two version of the experiment is developed for comparison purpose based on modularity and time taken. The results are shown in the tables 3.3 and 3.4 respectively.

Experimental steps are as follows:

**Design of experiment:** In this experiment we have compared three groups of algorithms for network community detection with one based on nearest neighbor search, described above. Two version of the experiment is developed for comparison purpose based on modularity and time taken in seconds.

**Best of literature:** Regarding the three groups of algorithms; first group contain algorithms based on semi-definite programming and the second group contain algorithms

Table 3.2: Exp2: Experiment on approximation

Name	exact	0.1mtree	0.1lsh	0.2mtree	0.2lsh	0.3mtree	0.3lsh	0.4mtree	0.4lsh	0.5mtree	0.5lsh
Facebook	0.5472	0.5468	0.5462	0.5463	0.5452	0.5459	0.5441	0.5454	0.5431	0.5450	0.5421
Gplus	0.4056	0.4053	0.4049	0.4050	0.4042	0.4047	0.4035	0.4044	0.4028	0.4041	0.4021
Twitter	0.3709	0.3706	0.3701	0.3702	0.3693	0.3699	0.3685	0.3695	0.3677	0.3692	0.3669
Epinions1	0.1447	0.1446	0.1445	0.1445	0.1443	0.1445	0.1441	0.1444	0.1439	0.1443	0.1437
LiveJournal1	0.1458	0.1456	0.1454	0.1455	0.1450	0.1453	0.1447	0.1452	0.1443	0.1450	0.1439
Pokec	0.0295	0.0294	0.0293	0.0294	0.0292	0.0293	0.0290	0.0293	0.0289	0.0292	0.0287
Slashdot0811	0.0125	0.0124	0.0123	0.0123	0.0122	0.0121	0.0120	0.0120	0.0119	0.0119	0.0117
Slashdot0922	0.0168	0.0167	0.0167	0.0167	0.0166	0.0166	0.0164	0.0166	0.0163	0.0165	0.0162
Friendster	0.0536	0.0535	0.0534	0.0534	0.0532	0.0533	0.0529	0.0532	0.0527	0.0531	0.0525
Orkut	0.0690	0.0689	0.0688	0.0688	0.0685	0.0687	0.0683	0.0686	0.0680	0.0685	0.0678
Youtube	0.0936	0.0936	0.0935	0.0935	0.0934	0.0935	0.0932	0.0934	0.0931	0.0934	0.0930
DBLP	0.4215	0.4211	0.4206	0.4207	0.4197	0.4204	0.4189	0.4200	0.4180	0.4196	0.4171
Arxiv-AstroPh	0.5081	0.5077	0.5072	0.5073	0.5063	0.5069	0.5053	0.5065	0.5044	0.5061	0.5035
web-Stanford	0.3908	0.3904	0.3900	0.3901	0.3891	0.3897	0.3883	0.3894	0.3874	0.3890	0.3866
Amazon0601	0.2650	0.2647	0.2644	0.2645	0.2638	0.2642	0.2633	0.2640	0.2627	0.2637	0.2621
P2P-Gnutella31	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523
RoadNet-CA	0.1692	0.1690	0.1686	0.1687	0.1681	0.1685	0.1675	0.1682	0.1670	0.1680	0.1664
Wiki-Vote	0.2095	0.2094	0.2093	0.2093	0.2090	0.2092	0.2088	0.2091	0.2085	0.2090	0.2083

based on graph traversal approaches. For each group, we have taken the best value of modularity in table 3.3 among all the algorithms in the groups. All the algorithms considered in this experiment are detailed in sec. 3.5.

**Other distance based methods:** Three different methods of network community detection are also considered for our comparison which indirectly use the influence between the nodes in their algorithms. These methods are walktrap(WT), label propagation(LP) and geometric brownian motion(GBM) and already discussed in section 3.5 along with their references and complexities.

**Proposed methods:** Three version of proposed algorithm is compared with other algorithms, the proposed algorithm based on exact nearest neighbor, approximated nearest neighbor computed using metric tree and approximate nearest neighbor computed using locality sensitive hashing.

**Network data:** A long list of real network data is taken for evaluation of modularity and time described in table 3.2.

**Efficiency and time:** Compared the community structure obtained in terms of modularity and time (seconds) taken by the algorithms, shown in the tables 3.3 and 3.4 respectively.

6) The results obtained with our approach are very competitive with most of the well known algorithms in the literature and this is justified over the large collection of datasets. On the other hand, it can be observed that time (second) taken (table 3.4) by our algorithm is quite less compared to other methods and justify the theoretical findings.

Table 3.3: Comparison of our approaches with other best methods in terms of modularity

Name	Spectral	SDP	GT	WT	LP	GBM	NN-search	M-tree	LSH
Facebook	0.4487	0.5464	0.5434	0.5117	0.5042	0.4742	0.5472	0.5450	0.5421
Gplus	0.2573	0.4047	0.3998	0.3528	0.3412	0.2963	0.4056	0.4041	0.4021
Twitter	0.3261	0.3706	0.3691	0.3545	0.3513	0.3375	0.3709	0.3692	0.3669
Epinions1	0.0280	0.1440	0.1401	0.1034	0.0940	0.0589	0.1447	0.1443	0.1437
LiveJournal1	0.0791	0.1455	0.1432	0.1220	0.1169	0.0966	0.1458	0.1450	0.1439
Pokec	0.0129	0.0294	0.0288	0.0235	0.0223	0.0172	0.0295	0.0292	0.0287
Slashdot0811	0.0038	0.0130	0.0127	0.0095	0.0090	0.0060	0.0125	0.0119	0.0117
Slashdot0922	0.0045	0.0176	0.0171	0.0127	0.0119	0.0078	0.0168	0.0165	0.0162
Friendster	0.0275	0.0536	0.0526	0.0443	0.0423	0.0343	0.0536	0.0531	0.0525
Orkut	0.0294	0.0689	0.0675	0.0549	0.0519	0.0398	0.0690	0.0685	0.0678
Youtube	0.0096	0.0934	0.0903	0.0640	0.0573	0.0319	0.0936	0.0934	0.0930
DBLP	0.4011	0.4214	0.4207	0.4136	0.4125	0.4060	0.4215	0.4196	0.4171
Arxiv-AstroPh	0.4174	0.5079	0.5045	0.4755	0.4688	0.4410	0.5081	0.5061	0.5035
web-Stanford	0.3595	0.3908	0.3896	0.3791	0.3772	0.3673	0.3908	0.3890	0.3866
Amazon0601	0.1768	0.2649	0.2615	0.2336	0.2269	0.1999	0.2650	0.2637	0.2621
P2P-Gnutella31	0.0009	0.0522	0.0503	0.0343	0.0301	0.0146	0.0523	0.0523	0.0523
RoadNet-CA	0.0212	0.1690	0.1633	0.1168	0.1053	0.0603	0.1692	0.1680	0.1664
Wiki-Vote	0.0266	0.2093	0.2023	0.1451	0.1306	0.0752	0.2095	0.2090	0.208



Table 3.4: Comparison of our approaches with other best methods in terms of time

Name	Spectral	SDP	GT	WT	LP	GBM	NN-search	M-tree	LSH
Facebook	6	7	11	13	7	8	6	4	1
Gplus	797	832	1342	1512	877	948	661	390	115
Twitter	462	485	786	886	509	554	398	235	68
Epinions1	411	419	667	749	452	475	292	174	56
LiveJournal1	1297	1332	2129	2394	1427	1514	969	576	179
Pokec	1281	1305	2075	2330	1410	1480	901	538	173
Slashdot0811	552	561	891	1000	608	636	382	228	74
Slashdot0922	561	570	906	1017	618	647	389	232	75
Friendster	2061	2105	3352	3766	2269	2390	1477	880	280
Orkut	1497	1529	2435	2736	1647	1735	1074	640	203
Youtube	829	844	1340	1505	913	957	578	345	111
DBLP	381	403	655	739	420	461	341	201	57
Arxiv-AstroPh	217	230	375	423	239	263	197	116	33
web-Stanford	498	525	852	960	549	600	437	258	74
Amazon0601	653	678	1089	1225	719	771	520	308	93
P2P-Gnutella31	182	184	293	328	200	209	124	74	24
RoadNet-CA	758	785	1261	1419	834	894	599	355	107
Wiki-Vote	54	55	88	99	59	63	39	23	7

### 3.6.6 Results analysis and achievements

In this subsection, we have described the analysis of the results obtained in our experiments shown. The results obtained in the first experiment justify that the proposed distance is more useful for complex network to extract the community structure compared to other measures of similarity. The results obtained in the second experiment verify that the approximate distance is also useful for network community detection especially for large data where time is a major concern. The results obtained in the third experiment justify that the proposed algorithm for community detection is very efficient compared to other existing methods in terms of modularity and time.

## 3.7 Conclusions

In this chapter, we have studied the interesting problem of the nearest neighbor within the nodes of a complex networks and applied this for community detection. We have used geometric framework for network community detection instead of traditional graph theoretic approach or spectral methods. Processing the nearest neighbor search in complex networks cannot be achieved straightforward; we presented the transformation of graph to metric space and efficient computation of the nearest neighbor therein using metric tree and locality sensitive hashing. To validate the performance of proposed nearest neighbor search designed for complex networks we applied our approaches on community detection problem. Several experiments conducted in this regard and we found community detection using nearest neighbor search is very efficient and time saving for large networks due to good approximations. The results obtained on several network data sets prove the usefulness of the proposed method and provide motivation for further application of other

## Chapter 4

# Low rank approximations for community detection in large complex networks

In this chapter we have proposed an algorithm to reduce the dimensionality induced matrices of large complex network via Nystrom sampling and random sketches, such that after applying partitioning algorithm on reduced network we get almost same results of community detection as the original network. In this work we will explore the two low rank approximation methods, Nystrom sampling and random sketching and they are described below. Low-rank methods are very fast to compute communities from massive networks by preserving the approximated results with moderate lower bound. For evaluation of the proposed approaches on complex network we applied it in community detection problem. The results obtained using our methods are very competitive with most of the well known algorithms exists in the literature and this is verified on collection of real networks. On the other-hand, it can be observed that time taken by our algorithm is quite less compared

to popular methods.

## 4.1 Introduction

Low rank approximation is a fundamental result of linear algebra. For any matrix  $A$  and positive integer  $k$ , low rank approximation method find a matrix  $A_k$  of rank  $k$  by minimizing  $\|A - A_k\|$ . The existence of such an optimal rank  $k$  approximation, denoted by  $A_k$ , and its efficient computation, follow from the Singular Value Decomposition of  $A$ , a manner of writing  $A$  as a sum of decreasingly significant rank one matrices. Long in the purview of numerical analysts, low rank approximations have recently gained broad popularity in computer science. For example, in areas such as computer vision, information retrieval, and machine learning they are used as a basic tool for extracting correlations and removing noise from matrix-structured data. However, applications of this technique to massive matrices, such as those arising from web corpora and extended video sequences, quickly run against practical computational limits. Specifically, orthogonal iteration and Lanczos iteration, the two most common algorithms for computing optimal low rank approximations, operate through repeated matrix-vector multiplication, thereby requiring superlinear time and large working sets.

In this chapter, we will study community detection of large complex network in the framework of low rank approximation. Detecting clusters or communities in real-world graphs such as large social networks, web graphs, and biological networks is a problem of considerable practical interest that has received a great deal of attention [56, 66]. A network community (also sometimes referred to as a module or cluster) is typically thought of as a group of nodes with more and/or better interactions amongst its members than between its members and the remainder of the network [29]. The objective is typically

NP-hard to optimize exactly [54, 74], one employs heuristics [18, 19] or approximation algorithms [74] to find sets of nodes that approximately optimize the objective function and that can be understood or interpreted as real communities. Community detection in real networks aims to capture the structural organization of the network using the connectivity information as input [54, 74]. Most of the methods developed for network community detection are based on a two-step approach. The first step is specifying a quality measure (evaluation measure, objective function) that quantifies the desired properties of communities and the second step is applying an algorithmic techniques to assign the nodes of graph into communities by optimizing the objective function.

For all these methods, finding the exact solution requires optimizing a function of the adjacency matrix  $A$  over  $(K, n)$  dimensional label matrix, which is an exponential optimization problem. In another line of work, spectral decompositions have been used in various ways to obtain approximate solutions that are much faster to compute. One such algorithm is spectral clustering, a generic clustering method which became popular for community detection. In spectral clustering, typically one first computes the normalized Laplacian matrix  $L = D^{-1/2}AD^{-1/2}$ , where  $D$  is a diagonal matrix with diagonal entries being node degrees  $d_i$ . Then the  $K$  eigenvectors of the Laplacian corresponding to the first  $K$  largest eigenvalues are computed, and their rows clustered using  $K$ -means into  $K$  clusters corresponding to different labels. It has been shown that spectral clustering performs better with further regularization, namely if a small constant is added either to  $D$  or to  $A$ .

In this work, we have tried to develop the notion of approximate distance among the nodes using some new matrices derived from adjacency matrix and degree matrix of the graph. Let  $A$  be the adjacency matrix and  $D$  the degree matrix of the graph  $G = (V, E)$ . The Laplacian  $L = D - A$ . We have defined two diagonal matrix of same size  $D(\lambda)$  and

$D(\lambda_x)$  where  $\lambda$  is a parameter determine from the given graph and can be optimized from the optimization criteria of the problem under consideration. In  $D(\lambda)$  a fixed optimally determine value is used in the diagonal entries of the matrix  $D$  and in  $D(\lambda_x)$  a variable value also optimally determine is used in the diagonal entries of the matrix  $D$ .

A low rank approximation is performed on the matrices  $L_1$  and  $L_2$ , where  $L_1 = D(\lambda) + A$  and  $L_2 = D(\lambda_x) + A$  respectively using nystrom sampling and random sketching. The spherical similarity among the rows and determine by applying a concave function  $\phi$  over the standard notions of similarities like, Pearson coefficient( $\sigma_{PC}$ ), Spacerman coefficient( $\sigma_{SC}$ ) or Cosine similarity( $\sigma_{CS}$ ).  $\phi(\sigma)()$  must be chosen using the chord condition to obtain a metric.

The rest of this paper is organized as follows: Section 4.3 described the problem of computing low rank distance between the nodes of the large complex network. In Section 4.3.4, the algorithm of approximate network community detection is formulated and the initialization procedures, termination criteria, convergence are discussed in detail. The results of comparison between community detection algorithms are illustrated in Section ???. The computational aspects of the proposed framework are also discussed in this section.

## 4.2 Low rank approximation of matrices

Traditional methods in information retrieval and machine learning deal with data in vectorized representation. A collection of data is then stored in a single matrix  $A \in R^{n \times m}$ , where each column of  $A$  corresponds to a vector in the  $n$  dimensional space. A major benefit of this vector space model is that the algebraic structure of the vector space can be exploited. However, crucial information intrinsic in the data should not be removed

under this simplification. A widely used method for this purpose is to approximate the single data matrix,  $A$ , with a matrix of lower rank. Mathematically, the optimal rank- $k$  approximation of a matrix  $A$ , under the Frobenius norm can be formulated as follows: Find a matrix  $B \in R^{n \times m}$  with  $\text{rank}(B) = k$ , such that  $\|A - B\|$  is minimum. The matrix  $B$  can be readily obtained by computing the Singular Value Decomposition (SVD) of  $A$ , as stated by Golub and Van Loan, 1996. For any approximation  $M$  of  $A$ , we call  $\|A - M\|_F$  the reconstruction error of the approximation.

Computing a low-rank approximation using the SVD is appealing from a theoretical point of view, since it provides the closest matrix with a given rank. For many applications where the data matrix is large, calculating the SVD can be impractical since it requires a large number of operations and it has large memory requirements. Recent research has thus focused on algorithms that are suboptimal, in the sense that the low-rank matrices that they calculate are not the closest possible to the original matrix. The advantage of using such algorithms is that they are faster than the SVD-based algorithm and need less memory, making them much more suitable for large scale applications.

Low-rank approximations have found numerous applications in various fields. Examples include Latent Semantic Indexing, Support Vector Machine training, Computer Vision, and Web Search models. In this applications, the data consist of a matrix of pairwise distance between the nodes of a complex network and approximated by a low-rank matrix for fast community detection using distance based partitioning algorithm. Calculating such a low-rank approximation can reveal the underlying structure of the data and allow for fast computations.

## 4.3 Network community detection using low rank approximation

In this section we have proposed an algorithm to reduce the dimensionality via Nystrom sampling and random sketches, such that after applying partitioning algorithm on reduced network we get almost same results as with the original network.

In this work we will explore the two low rank approximation methods, Nystrom sampling and random sketching and they are described below. Low-rank methods are very fast to compute communities from massive networks by preserving the approximated results with moderate lower bound.

We used the approximate distance between nodes for network community detection with the help of distance based partitioning algorithm.

### 4.3.1 Distance matrix of complex network

In this section we have demonstrated the algorithm to convert the nodes of the graph to the points of a metric space preserving the community structure of the graph. The algorithm depends on the sub modules 1) construction of  $L_x$  ( $L_1$  or  $L_2$ ) 2) approximating  $L_x$  by low rank methods, and 3) obtaining a structure preserving distance function. The algorithm works as picking pair of nodes from  $L_x$  and computing distance defined in the second module.

#### $L_x$ construction

The  $L_1$  is defined as  $L_1 = D(\lambda) + A$ , where  $A$  is the adjacency matrix of the given network and  $D(\lambda)$  is a diagonal matrix of same size with diagonal values equal to a non negative



constant  $\lambda$ .

The  $L_2$  is defined as  $L_2 = D(\lambda_x) + A$ , where  $A$  is the adjacency matrix of the given network and  $D(\lambda_x)$  is a diagonal matrix of same size with diagonal values determine by a non negative function  $\lambda_x$  of the node  $x$ .

The choice of  $\lambda$  and  $\lambda_x$  plays a crucial role in combination with the function chosen in the second module for determination of a suitable metric and is discussed later part of this subsection.

### Function selection

The function selection module determine the metric for pair of nodes. The function selector  $\phi$  converts a similarity function (Pearson coefficient( $\sigma_{PC}$ ), Spacerman coefficient( $\sigma_{SC}$ ) or Cosine similarity( $\sigma_{CS}$ )) into a distance matrix. In general the similarity function satisfies the positivity and similarity condition of metric but not triangle inequality.

### Choice of $\lambda$ and $\phi(\sigma)()$

The choices in the above sub modules play a crucial role in the graph to metric transformation algorithm to be used for community detection. The complex network is characterized by small average diameter and high clustering coefficient. Several studies on network structure analysis reveal that there are hub nodes and local nodes characterizing the interesting structure of the complex network.

### 4.3.2 Nystrom approximation

The Nystrom methods approximate any SPSP matrix in terms of a subset of its columns. Specifically, given an  $m \times m$  SPSP matrix  $A$ , they require sampling  $c(< m)$  columns of  $A$

to construct an  $m \times c$  matrix  $C$ . We always assume that  $C$  consists of the first  $c$  columns of  $A$  without loss of generality. We partition  $A$  and  $C$  as

$$A = \begin{pmatrix} W & A_{21}^T \\ A_{21} & A_{22} \end{pmatrix} \text{ and } C = \begin{pmatrix} W \\ A_{21} \end{pmatrix}$$

where  $W$  and  $A_{21}$  are of size  $c \times c$  and  $m - c \times c$  respectively [43]

**Definition** Nystrom Approximation The standard Nystrom approximation of  $A$  is

$$A_c^{nys} = \begin{pmatrix} W & A_{21}^T \\ A_{21} & A_{21}W & A_{21}^T \end{pmatrix}$$

Since the running time complexity of SVD on  $W$  is in  $O(kl^2)$  and matrix multiplication with  $C$  takes  $O(kln)$ , the total complexity of the Nystrom approximation computation is in  $O(kln)$  [2, 25].

### 4.3.3 Random Sketching

In random sketch, firstly relation is modelled as defining a vector or matrix, and then the sketch is formed by multiplying the data by a vector. Figure 1 is demonstrating it in which a fixed sketch matrix multiplies the data to generate the sketch (vector) [14]. Sketch vector forms the synopsis of the entire data, which is much smaller than the original data. Data mining algorithms can now be applied on the sketch vector.

**Definition** Johnson-Lindenstrauss Lemma 1 For any set of  $n$  points  $S \in R^d$ , there is a  $(1 + \epsilon)$ -distortion embedding of  $X$  into  $R^k (k < d)$ , for  $k = O(\frac{\log(n)}{\epsilon^2})$

**Definition** Johnson-Lindenstrauss Lemma 2 There is a distribution over random linear mappings  $A : R^d \rightarrow R^k, (k < d)$  such that for any vector  $x$  we have  $\|Ax\| = (1 \pm \epsilon)\|x\|$  with probability  $1 - e^{-Ck\epsilon^2}$

The Johnson Linderstrauss theorems [40] for sketches ensures the preservation of distance during lower dimensional approximation and provide minimum lower bound of approximation.

In this algorithm we have used random sketch to reduce the dimensionality of the dataset. Step 1 counts the number of attributes in the dataset and in step 2 sketch matrix is generated which have rows equal to the number attributed in the dataset. Sketch matrix have 3 sets of values (1) between 0 and 1 (2) 0 and 1 (3) between maximum and minimum value of the dataset. Step 3 multiplies each row of the dataset with the sketch matrix in order to generate sketch vector. Once the Sketch vector is generated K-Means, Hierarchical clustering and K-Nearest Neighbour algorithms are applied on the Sketch Matrix.

#### 4.3.4 Approximate community detection

In this section we have described k-central algorithm for the purpose of network community detection by using the nearest neighbor search inside complex network. We have also studied and analyzed the advantages of the k-central method over the standard algorithm for network community detection. Scalability is a major challenge faced by all the metric based algorithms, as they require computation of the full distance matrix whose size is quadratic in the number of data points. To the best of our knowledge, only a few attempts have been made to scale community detection algorithms to large data sets.

### Approximate k-central algorithm

The community detection methods based on partitioning of graph is possible using approximate nearest neighbor search, because the nodes of the graph are converted into the points of a metric space. A pairwise distance matrix is computed between the nodes of the network using the proposed metric. The low rank approximation is applied on the matrix of pairwise distance. Two type of approximations are considered here, Nystrom sampling and random sketching. A modified k center algorithm is used to detect communities of the network to use the approximate distance between nodes and to use approximate center of the partition. This algorithm for network community detection converges automatically and does not compute the value of objective function in iterations therefore reduce the computation compared to standard methods. The approximate k-central algorithm for community detection and its details analysis is given below.

---

**Algorithm 5** approximate k-central algorithm

---

**Require:**  $M = (V, d)$

**Ensure:**  $T = \{C_1, C_2, \dots, C_k\}$  with minimum  $cost(T)$

```
1: Initialize centers  $z_1, \dots, z_k \in R^n$  and clusters  $T = \{C_1, C_2, \dots, C_k\}$ 
2: repeat
3:   for  $i = 1$  to  $k$  do
4:     for  $j = 1$  to  $k$  do
5:        $C_i \leftarrow \{x \in V \text{ s.t. } |z_i - x| \leq |z_j - x|\}$ 
6:     end for
7:   end for
8:   for  $j = 1$  to  $k$  do
9:      $z_i \leftarrow Central(C_i)$  ; where  $Central(C_i)$  returns the node with minimum total
       distance in the class of consideration.
10:  end for
11: until  $|cost(T_t) - cost(T_{t+1})| = 0$ 
12: return  $T = \{C_1, C_2, \dots, C_k\}$ 
```

---

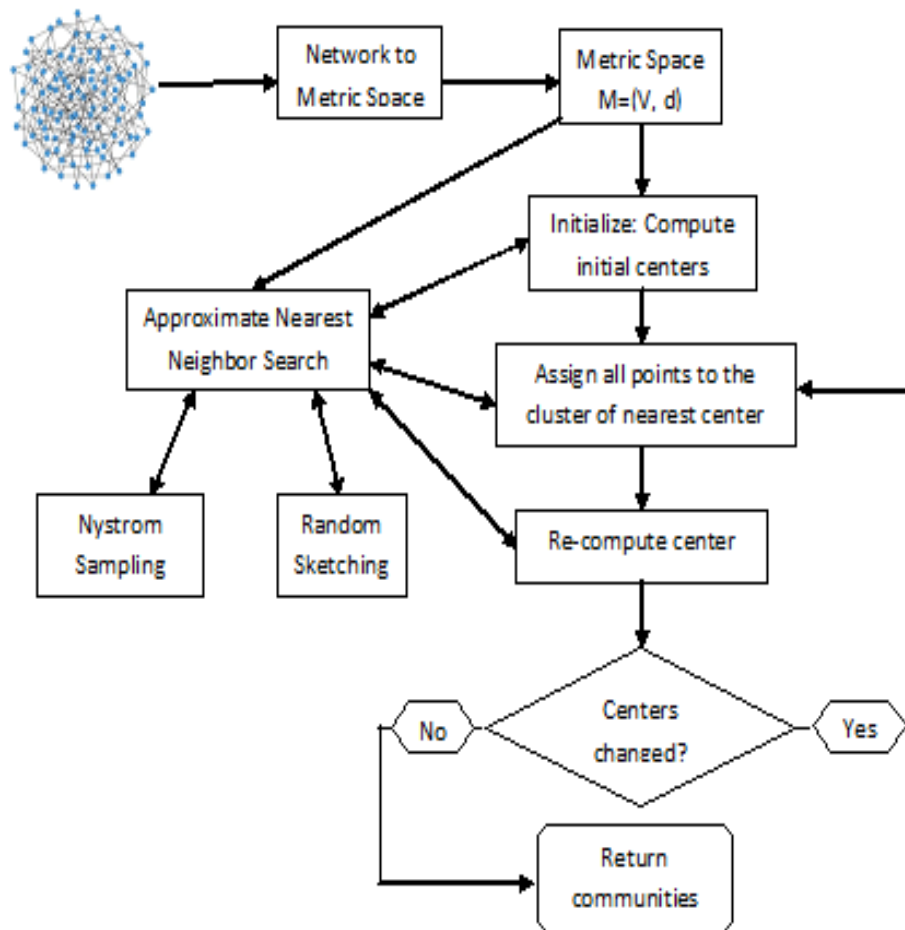


Figure 4.1: Block diagram of approximate k-central algorithm using om lowrank approximation with Nystrom sampling or random sketching

## 4.4 Experiments and results

We performed many of experiments to test the performance of low rank approximation based community detection method for complex network over several real networks 2.2. Objective of the experiment is to verify behavior of the algorithm and the time required to compute the algorithm. One of the major goals of the experiment is to verify the behavior of the algorithm with respect to the performance of other popular methods exists in the literature with respect to the modularity.

### 4.4.1 Experimental designs

Experiment for comparison: In this experiment we have compared several algorithms for network community detection with our proposed algorithm developed using nearest neighbor search in complex network. Experiment is performed on a large list of network data sets. Two version of the experiment is developed for comparison purpose based on modularity and time. The results are shown in the table 4.1. In the first experiment we have evaluated our algorithm for performance on the network collection 2.2 in terms of modularity. In the second experiment we have evaluated the time taken by our algorithm on different size of networks and is shown in the table 4.2.

### 4.4.2 Computational results

In this subsection we have compared two groups of algorithms for network community detection with our proposed algorithm using nearest neighbor search. Experiment is performed on a large list of network data sets. Regarding the two groups of algorithms; first group contain algorithms based on semi-definite programming and the second group

contain algorithms based on graph traversal approaches. For each group, we have taken the best value of modularity in table 4.1 among all the algorithms in the groups. The results obtained with our approach are very competitive with most of the well known algorithms in the literature and this is justified over the large collection of datasets. On the other hand, it can be observed that time taken (table 4.2) by our algorithm is very less compared to other methods and justify the theoretical findings.

Table 4.1: Comparison of our approaches with other best methods in terms of modularity

Name	Spectral	SDP	GT	k-center	Nystrom	R-sketches
Facebook	0.4487	0.5464	0.5434	0.5472	0.5450	0.5421
Gplus	0.2573	0.4047	0.3998	0.4056	0.4041	0.4021
Twitter	0.3261	0.3706	0.3691	0.3709	0.3692	0.3669
Epinions1	0.0280	0.1440	0.1401	0.1447	0.1443	0.1437
LiveJournal1	0.0791	0.1455	0.1432	0.1458	0.1450	0.1439
Pokec	0.0129	0.0294	0.0288	0.0295	0.0292	0.0287
Slashdot0811	0.0038	0.0130	0.0127	0.0131	0.0129	0.0127
Slashdot0922	0.0045	0.0176	0.0171	0.0176	0.0174	0.0172
Friendster	0.0275	0.0536	0.0526	0.0536	0.0531	0.0525
Orkut	0.0294	0.0689	0.0675	0.0690	0.0685	0.0678
Youtube	0.0096	0.0934	0.0903	0.0936	0.0934	0.0930
DBLP	0.4011	0.4214	0.4207	0.4215	0.4196	0.4171
Arxiv-AstroPh	0.4174	0.5079	0.5045	0.5081	0.5061	0.5035
web-Stanford	0.3595	0.3908	0.3896	0.3908	0.3890	0.3866
Amazon0601	0.1768	0.2649	0.2615	0.2650	0.2637	0.2621
P2P-Gnutella31	0.0009	0.0522	0.0503	0.0523	0.0523	0.0523
RoadNet-CA	0.0212	0.1690	0.1633	0.1692	0.1680	0.1664
Wiki-Vote	0.0266	0.2093	0.2023	0.2095	0.2090	0.2083

### 4.4.3 Results analysis and achievements

In this subsection, we have described the analysis of the results obtained in our experiments shown above and also highlighted the achievements from the results. It is clearly

Table 4.2: Comparison of our approaches with other best methods in terms of time

Name	Spectral	SDP	GT	k-center	Nystrom	R-sketches
Facebook	6	7	11	6	4	1
Gplus	797	832	1342	661	390	115
Twitter	462	485	786	398	235	68
Epinions1	411	419	667	292	174	56
LiveJournal1	1297	1332	2129	969	576	179
Pokec	1281	1305	2075	901	538	173
Slashdot0811	552	561	891	382	228	74
Slashdot0922	561	570	906	389	232	75
Friendster	2061	2105	3352	1477	880	280
Orkut	1497	1529	2435	1074	640	203
Youtube	829	844	1340	578	345	111
DBLP	381	403	655	341	201	57
Arxiv-AstroPh	217	230	375	197	116	33
web-Stanford	498	525	852	437	258	74
Amazon0601	653	678	1089	520	308	93
P2P-Gnutella31	182	184	293	124	74	24
RoadNet-CA	758	785	1261	599	355	107
Wiki-Vote	54	55	88	39	23	7

evident from the results shown in the tables 4.1 and 4.2 that, proposed nearest neighbor based method for network community detection using low rank approximation provide very good competitive performance with respect to modularity and time taken in seconds.

## 4.5 Conclusions

In this chapter, we have studied the interesting problem of approximate community detection in large complex networks. We presented the transformation of network to distance matrix and efficient computation of approximate nearest neighbor therein using low rank approximation. Our techniques can be applied for quick analysis of very large complex network. To validate the performance of proposed low rank approximate method for com-



plex networks we applied our approaches on community detection problem. The results obtained on several network data sets prove the usefulness of the proposed method and provide motivation for further application of other structural analysis of complex network.

üz

## Chapter 5

# Optimal consensus-community detection for complex network

In this chapter, we present a comparative analysis of communities detected by several algorithms to redefine new community structure using rough set theoretic framework. Our approach enables the assessment of the structural discernibility among the output of multiple community detection algorithms and between the output of algorithms and communities that arise in practice. We have created an information system of communities which are partitions on the set of nodes in a network. The outputs of different community detection algorithms constitutes super partition of the set of nodes. The tasks of consensus community detection is to determine a marged optimal partition, which provides minimum discernibility with others. To demonstrate this concept, we proposed an algorithm for marging super partition until it constitutes desired number of communities. To asses the quality of the extracted community, we further extended the information system of communities into a decision system by considering the newly extracted communities as the decision attribute and computed the support and confidence of each partition of the

decision system.

Applied to a diverse collection of large scale network datasets, the analysis reveals that (1) the different detection algorithms extract different structures with respective constrained; (2) the structure of communities that arise in our method is closest to that of communities of other algorithms in terms of mutual indecernibility; and (3) consensus communities extracted from heterogeneous community detection algorithms determine much better meta community of the same network in terms of modularity.

## 5.1 Introduction

Community structure captures the tendency of entities in a network to group together in meaningful subsets whose members have a distinctive relationship to one another. The identification of these subsets allows for the analysis of networks at different levels of detail, which is instrumental in illuminating the structure underlying large scale systems.

Despite playing a fundamental role in the structure and function of networks, community structure has proved to be very difficult to define, quantify, and extract. In addition to challenges related to computational tractability, three major factors account for the intricacies of community extraction. First, the application domain includes a wide variety of networks of fundamentally different natures. Each of these networks possesses meaningful communities that may possess their own distinctive structural profiles. Second, the literature offers a multitude of disparate community detection algorithms. Due to differences in concept and design, the output of these procedures exhibits high structural variability across the collection. Last, there is no established consensus on the question of what properties distinguish subgraphs that are communities from those that are not communities. In this chapter, we present a comparative analysis of communities detected by

several algorithms to redefine new community structure using rough set theoretic framework. Our approach enables the assesment of the structural discernibility among the output of multiple community detection algorithms and between the output of algorithms and communities that arise in practice. We have created an information system of communities which are partitions on the set of nodes in a network. The outputs of different community detection algorithms constitutes super partition of the set of nodes. The tasks of consensus community detection is to determine a marged optimal partition, which provides minimum discernibility with others. To demonstrate this concept, we proposed an algorithm for marging super partition until it constitutes desired number of communities. To asses the quality of the extracted community, we further extended the information system of communities into a decision system by considering the newly extracted communities as the decision attribute and computed the support and confidence of each partition of the decision system. The combination of communities determined by different algorithms are analyzed to decide minimal reduct. Now from the minimal reduct obtained in the previous step we compute optimal consensus communities nusing rough set based indercinibility measure. In order to realize the specified objectives, the paper introduces rough set preliminaries in Section 5.2.1. Section 5.3 presents algorethmic foundation of the proposed method to extract optimal communities in rough set paradigm. Section 5.4 covers theoretical results regarding the quality and optimality of the extracted consensus communities. Finally, the performance and ecpperimental results of the algorithm are reported in Section 5.5.

## 5.2 Rough Set Theory

### 5.2.1 Introduction to Rough Set Theory

Rough set theory was developed by Zdzislaw Pawlak in the early 1980's [61]. It deals with the classificatory analysis of data tables. The data can be acquired from measurements or from human experts. The main goal of the rough set analysis is to synthesize approximation of concepts from the acquired data. We initially describe how synthesis takes place in an information system. In some instances, the aim may be to gain insight into the problem at hand by analyzing the constructed model, i.e., the structure of the model is itself of interest. In other applications, the transparent and explainable features of the model may be of secondary importance and the main objective is to construct a classifier that classifies unseen objects well. An important feature of rough sets is that the theory is followed by practical implementations of toolkits that support interactive model development [77].

#### Information systems and indiscernibility

A complete information system expresses all the knowledge available about the objects being studied. More formally, an information system is a pair,  $S = (U, A)$ , where  $U$  is a non-empty finite set of objects called the universe and  $A = \{a_1, a_2, \dots, a_j\}$  is a non-empty finite set of attributes on  $U$ . With every attribute  $a \in A$  we associate a set  $V_a$  such that  $a : U \rightarrow V_a$ . The set  $V_a$  is called the value set of  $a$  [61, 62]. This value set equates to the range of values associated with a specific variable. The data set  $U$  contained in the information system is used as the basis for the development of subsets of it that are “coarser” than  $U$ . As with any data analysis technique, details are lost,

but the removal of details are controlled to uncover the underlying characteristics of the data. The technique works by, lowering the degree of precision in data, based on a rigorous mathematical theory. A core concept of rough sets theory is that of equivalence between objects (called indiscernibility). Objects in the information system about which we have the same knowledge form an equivalence relation. If  $B \subset A$  there is an associated equivalence relation,  $IND_A(B)$ , called the B-indiscernibility relation. It is defined as:  $IND_A(B) = \{(x, \acute{x}) \in U^2 \mid \forall a \in B, a(x) = a(\acute{x})\}$ . If  $(x, \acute{x}) \in IND_A(B)$ , then the objects  $x$  and  $\acute{x}$  are indiscernible from each other when considering the subset  $B$  of attributes. Equivalence relations lead to the universe being divided into partitions, which can then be used to build new subsets of the universe [63, 77].

### Lower and upper approximations

Let  $S = (U, A)$  be an information system, and let  $B \subset A$  and  $X \subset U$ . We can describe the subset  $X$  using only the information contained in the attribute values from the subset  $B$  by constructing two subsets, referred to as the B-lower and B-upper approximations of  $X$ , and denoted as  $B_*(X)$  and  $B^*(X)$  respectively, where:  $B_*(X) = \{x \mid [x]_B \in X\}$ , where  $[x]_B$  is an equivalence class corresponding to  $B$  and  $B^*(X) = \{x \mid [x]_B \cap X \neq \emptyset\}$ , where  $[x]_B$  is an equivalence class corresponding to  $B$ . The lower approximation contains objects that are definitely in the subset  $X$  and the upper approximation contains objects that may or may not be in  $X$ . A third subset is also useful in analysis, the boundary region, which is the difference between the upper and lower approximations. This definition of a rough (approximate) set in terms of two other sets is contributed by Pawlak [61–63]. Any partition  $P$  of universe  $U$  defines an indiscernibility relation  $IND(P) : xIND(P)y$  iff  $(x, y \in X)$  for some  $X \in P$ . Let  $P = \{P_1, P_2, \dots, P_n\}$ ,  $Q = \{Q_1, Q_2, \dots, Q_m\}$  are partitions of  $U$ . We define the  $P$ -lower approximation of  $Q$  and the  $P$ -upper approximation of  $Q$ , respectively

by  $P_*Q = \{P_*Q_1, P_*Q_2, \dots, P_*Q_m\}$  where  $P_*Q_i = \{x \in U : x \in P_j \subseteq Q_i \text{ for some } P_j \in P\}$  for  $i = 1, 2, \dots, m$   $P^*Q = \{P^*Q_1, P^*Q_2, \dots, P^*Q_m\}$  where  $P^*Q_i = \{x \in U : x \in P_j \text{ and } P_j \cap Q_i \neq \emptyset \text{ for some } P_j \in P\}$  for  $i = 1, 2, \dots, m$ .

## Reduct and core

In the previous section we investigated one natural dimension of reducing data which is to identify equivalence classes, i.e., objects that are indiscernible using the available attributes. Savings are to be made since only one element of the equivalence class is needed to represent the entire class. The other dimension in reduction is to keep only those attributes that preserve the indiscernibility relation and, consequently, set approximation. The rejected attributes are redundant since their removal cannot worsen the partition. There is usually several such subsets of attributes and those which are minimal are called reducts. Computing equivalence classes is straightforward. Finding a minimal reduct (i.e., reduct with a minimal cardinality of attributes among all reducts) is NP-hard. One can also show that the number of reducts of an information system with  $m$  attributes may be equal to  $m$  choose floor of  $m/2$ . This means that computing reducts it is a non-trivial task that cannot be solved by a simple minded increase of computational resources. It is, in fact, one of the bottlenecks of the rough set methodology. Fortunately, there exist good heuristics (e.g., [GENETIC REDUCT]) based on genetic algorithms that compute sufficiently many reducts in often acceptable time, unless the number of attributes is very high.

## Decision rules

To date, most of the published literature in rough sets has concentrated on a specific type of information system, referred to as a decision system. In a decision system, at

least one of the attributes is a decision attribute. This decision attribute partitions the information system into concepts. The rule generation problem is expressed in rough set theory as finding mappings from the partitions induced by the equivalence relations in the condition attributes to the partitions induced by the equivalence relations in the decision attribute(s). These mappings are usually expressed in terms of decision rules. More formally we can associate a formal language  $L(S)$  with an information system  $S = (U, A)$ . Expressions in this language are logical formulas built up from attributes and attribute-value pairs and standard logical connectives (Pawlak 1999). A decision rule in  $L$  is an expression  $P \rightarrow Q$  (read if  $P$  then  $Q$ ), where  $P$  and  $Q$  are respectively the conditions and decisions of the rule. Each rule can be assigned a confidence factor, which is the number of objects in the attribute subset that also satisfy the decision subset (concept), divided by the total number of objects in the attribute subset.

### 5.3 Optimal community detection using rough set

Our algorithm to extract consensus community is designed to work on the output of several community detection algorithms. Each community detection algorithm determine a partition on the set of nodes of the network. Collection of different partition constitutes a information system for the nodes of the network. Communities determined by the individual algorithms are condition attributes of the information system. Our algorithm analyse the information system to extract a suitable community structure of the network with better quality then its constituents. Rough set based attribute reduction is performed to determine core and minimal reduct. The attributes corresponding to minimal reduct are then taken to extract the optimal communities.



### 5.3.1 Description of method

#### Meta data generation

The outputs of the several community detection algorithms are required to create information system for rough set theoretic analysis. The information system is created by meta data because they are generated as output of the algorithms on input network. This meta data represented in the form of information system is the input of rough set data analysis task. Unlike the complex network, meta data has a simple brief format, where communities determined by the respective algorithm contribute the existence of an attribute, values of this attribute can be a tag associated with a particular community determined by the algorithm. So the number of attributes is the same as number of algorithms and the number of objects is equal to the number of nodes in the actual network.

#### Analysis of meta data

Rough set based attribute reduction techniques eliminate superfluous attributes and create a minimal sufficient subset of attributes. Such minimal sufficient subset of attributes, called a reduct, is an essential part which discern all examples discernible by the original table and cannot be reduced any more. Given set of algorithms may have more than one reduced set, all of which perform same as original. Once the reduct is computed redundant for the further analysis are removed from the information system.

#### Extracting communities from meta data

In this step rough set theoretic analysis to extract meta community is described. The outputs of different community detection algorithms constitutes super partition of the set of nodes. The tasks of consensus community detection is to determine a marged optimal

partition, which provides minimum discernibility with others. To demonstrate this concept, we proposed an algorithm for merging super partition until it constitutes desired number of communities. The algorithms are given in 6 and 7.

---

**Algorithm 6** Optimal community detection using rough discernibility

---

**Require:**  $G = (V, E)$ ,  $\Pi_1, \Pi_2, \Pi_3, \Pi_4, \dots, \Pi_k$

**Ensure:**  $\Pi_o$  - the optimal community

- 1: Let  $\Pi_s$  be the super-partition of  $\Pi_1, \Pi_2, \Pi_3, \Pi_4, \dots, \Pi_k$
  - 2: Find  $\Pi_x \subset \Pi_s$  s.t.  $\min_{i=1}^k \{|\Pi_i|\} \leq |\Pi_x| \leq \max_{i=1}^k \{|\Pi_i|\}$
  - 3:  $\Pi_o = \Pi_x$  with minimum discernibility
  - 4: **return**  $\Pi_o$
- 

### 5.3.2 Consensus community determination

In the notion of rough set, let  $U$  be the set of all nodes in the network and  $P^* = \{C_1, C_2, \dots, C_k\}$  where  $C_i \neq \phi$  for  $i = 1, 2, 3, \dots, k$ ,  $\cup_{i=1}^k C_i = U$  and  $C_i \cap C_j = \phi$  for  $i \neq j$  and  $i, j = 1, 2, 3, \dots, k$  be the extracted consensus communities of  $U$  which determines the categories of  $U$ . Output of a community detection algorithm determines a new partition on  $U$ . In rough set terminology each class of the given partition is a given concept about node set and the output of algorithms determines new concepts about the same data set. Now extracted concepts can be expressed approximately by upper approximation and lower approximation constructed by the generated concepts.

**Example:** Let  $S = \{1, 2, 3, \dots, 100\}$  be a set with an optimal partition determined.  $P = \{P_{01} = \{1, 2, \dots, 20\}, P_{02} = \{21, 22, \dots, 40\}, P_{03} = \{41, 42, \dots, 60\}, P_{04} = \{61, 62, \dots, 80\}, P_{05} = \{81, 82, \dots, 100\}\}$

Let algorithm  $a_1$  generate a partition  $P_1 = \{P_{11} = \{1, 2, \dots, 10\}, P_{12} = \{21, 22, \dots, 40\}, P_{13} = \{41, 42, \dots, 60\}, P_{14} = \{61, 62, \dots, 80\}, P_{15} = \{11, 12, \dots, 20, 81, 82, \dots, 100\}\}$ , algorithm  $a_2$  generate a partition  $P_2 = \{P_{21} = \{1, 2, \dots, 20\}, P_{22} =$

Table 5.1: Expressing  $P$  by lower approximation and upper approximation of other partitions

	$P_{01}$	$P_{02}$	$P_{03}$	$P_{04}$	$P_{05}$
$P_{1*}$	$P_{11}$	$P_{12}$	$P_{13}$	$P_{14}$	$\phi$
$P_1^*$	$P_{11} \cup P_{15}$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15}$
$P_{2*}$	$P_{21}$	$P_{22}$	$P_{23}$	$P_{24}$	$\phi$
$P_2^*$	$P_{21}$	$P_{22}$	$P_{23}$	$P_{24} \cup P_{25}$	$P_{25}$
$P_{3*}$	$P_{31}$	$P_{32}$	$P_{33}$	$P_{34}$	$\phi$
$P_3^*$	$P_{31} \cup P_{35}$	$P_{32} \cup P_{35}$	$P_{33} \cup P_{35}$	$P_{34} \cup P_{35}$	$P_{35}$
$(P_1 \cap P_2)_*$	$P_{21}$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15} \cap P_{25}$
$(P_1 \cap P_2)^*$	$P_{21}$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15} \cap P_{25}$
$(P_1 \cap P_3)_*$	$P_{31}$	$P_{12}$	$P_{13}$	$P_{14}$	$\phi$
$(P_1 \cap P_3)^*$	$P_{11} \cup P_{15}$	$P_{12}$	$P_{13}$	$P_{14}$	$P_{15} \cap P_{35}$
$(P_2 \cap P_3)_*$	$P_{21}$	$P_{22}$	$P_{23}$	$P_{34}$	$\phi$
$(P_2 \cap P_3)^*$	$P_{21}$	$P_{22}$	$P_{23}$	$P_{24} \cup P_{25}$	$P_{25} \cap P_{35}$

$\{21, 22, \dots, 40\}$ ,  $P_{23} = \{41, 42, \dots, 60\}$ ,  $P_{24} = \{61, 62, \dots, 70\}$ ,  $P_{25} = \{71, 72, \dots, 80, 81, 82, \dots, 100\}$ , and algorithm  $a_3$  generate a partition  $P_3 = \{P_{31} = \{1, 2, \dots, 19\}$ ,  $P_{32} = \{21, 22, \dots, 39\}$ ,  $P_{33} = \{41, 42, \dots, 59\}$ ,  $P_{34} = \{61, 62, \dots, 79\}$ ,  $P_{35} = \{20, 40, 60, 80, 81, 82, \dots, 100\}\}$ .

Error rates of algorithms  $a_1$ ,  $a_2$  and  $a_3$  are 10%, 10% and 5% respectively. The concept of  $P$  has been represented in terms of lower approximation and upper approximation by other partitions,  $P_1$ ,  $P_2$ ,  $P_3$ ,  $P_1 \cup P_2$ ,  $P_1 \cup P_3$  and  $P_2 \cup P_3$  in Table 5.1.

Since combination of  $P_1$  and  $P_2$  can express the given partition accurately, one does not need to use any other partition with  $P_1$  and  $P_2$ . If a case such as in the example is occurred, i.e., each set of  $P$  is defined by some partition  $P_i$ , where  $i = 1, 2, 3$  then one can use this fact in consensus community determination.

---

**Algorithm 7** Consensus community determination function

---

**Require:**  $\Pi_s$  be the super-partition of  $\Pi_1, \Pi_2, \Pi_3, \Pi_4, \dots, \Pi_k$

**Ensure:**  $\Pi_o = \min_{i=1}^k \{|\Pi_i|\} \leq |\Pi_x| \leq \max_{i=1}^k \{|\Pi_i|\}$ , with minimum discernibility

- 1: Let  $\Pi_s$  be the super-partition of  $\Pi_1, \Pi_2, \Pi_3, \Pi_4, \dots, \Pi_k$
  - 2: Select pair of partition with minimum indistinguishability and marge them.
  - 3: Replace both the partitions with their combined one.
  - 4: Continue till desired number of partition is reached.
  - 5: **return** Optimal partition
- 

### 5.3.3 Significance of input algorithms

One of the first ideas is to consider as relevant features those in the core of an information system, i.e., features that belong to the intersection of all reducts of the information system. It is also possible to consider as relevant features those from some approximate reducts of sufficiently high quality. As it follows from the considerations concerning reduction of attributes, they can be not equally important and some of them can be eliminated from an information table without losing information contained in the table. The idea of attribute reduction can be generalized by an introduction of the concept of significance of attributes, which enables an evaluation of attributes not only by a two-valued scale, dispensable - indispensable, but by associating with an attribute a real number from the  $[0,1]$  closed interval; this number expresses the importance of the attribute in the information table. Significance of an attribute  $a$  in an information system  $A = (U, C)$  can be evaluated by measuring the effect of removing of an attribute  $a \in C$  from the attribute set  $C$ . The number  $\gamma(a_i, a_j)$  expresses the degree of dependency between attributes  $a_i$  and  $a_j$ , or accuracy of approximation of  $U/a_i$  by  $a_j$ . The significance of an attribute  $a_i \in C$  with respect to  $a_j$  is define as the normalized difference between  $\gamma(C, a_j)$  and  $\gamma(C \setminus \{a_i\}, a_j)$ .

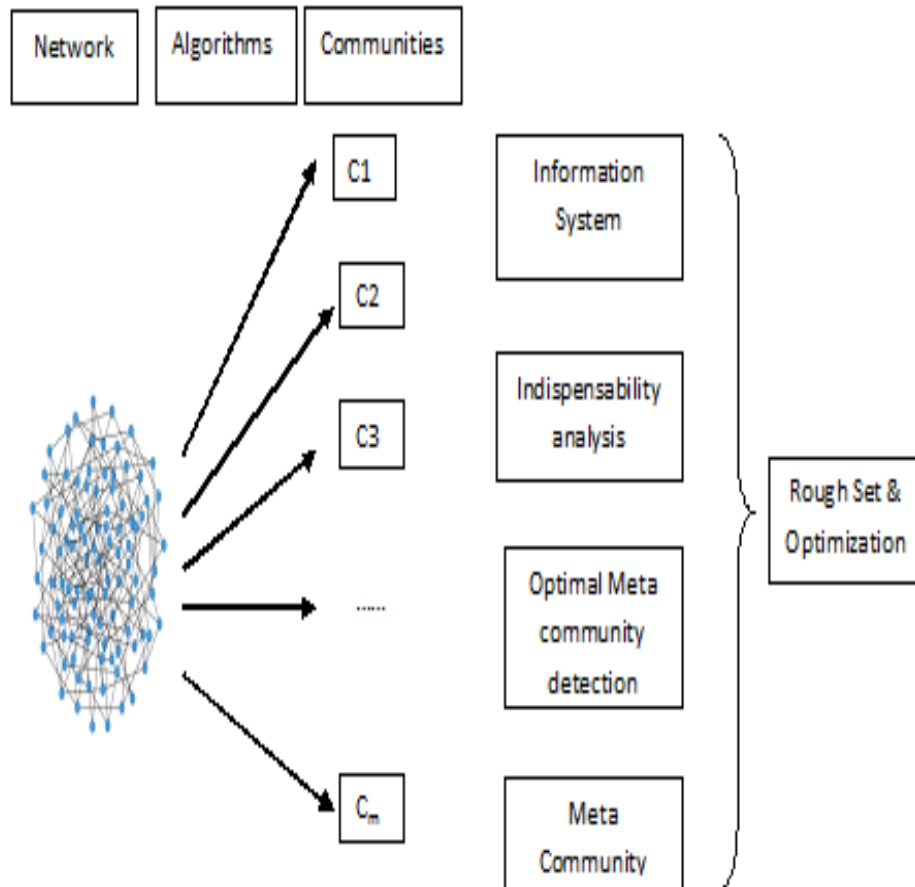


Figure 5.1: Block diagram of meta community detection algorithm using rough set theory and optimization

$$\sigma_{(C, a_j)}(a_i) = \frac{(\gamma(C, a_j) - \gamma(C \setminus \{a_i\}, a_j))}{\gamma(C, a_j)}$$

Thus the coefficient  $\sigma(a)$  can be understood as the error of representation which occurs when attribute  $a$  is dropped.

### 5.3.4 Relation between discernibility and diversity of input communities

The measure of the diversity of two partition used here is dependency between attribute at meta decision table. The smaller the dependency, the greater the diversity of the base-level partition. A set of attributes  $D$  depends totally on a set of attributes  $C$ , denoted  $C \Rightarrow D$ , if all values of attributes from  $D$  are uniquely determined by values of attributes from  $C$ . In other words,  $D$  depends totally on  $C$ , if there exists a functional dependency between values of  $D$  and  $C$ . Formally dependency can be defined in the following way. Let  $D$  and  $C$  be subsets of  $A$ . We will say that  $D$  depends on  $C$  in a degree  $k$  ( $0 \leq k \leq 1$ ), denoted  $C \Rightarrow_k D$ , if  $k = \gamma(C, D) = |POS_C(D)|/|U|$ , where  $POS_C(D) = \cup_{X \in U/D} \underline{C}(X)$ , called a positive region of the partition  $U/D$  with respect to  $C$ , is the set of all elements of  $U$  that can be uniquely classified to blocks of the partition  $U/D$ , by means of  $C$ . Obviously

$$\gamma(C, D) = \sum_{X \in U/D} \frac{|\underline{C}(X)|}{|U|}. \quad (5.1)$$

If  $k = 1$  we say that  $D$  depends totally on  $C$ , and if  $k < 1$ , we say that  $D$  depends partially (in a degree  $k$ ) on  $C$ . The coefficient  $k$  expresses the ratio of all elements of the universe, which can be properly classified to blocks of the partition  $U/D$ , employing attributes  $C$  and will be called the degree of the dependency. It can be easily seen that

if  $D$  depends totally on  $C$  then  $IND(C)$  subset  $IND(D)$ . This means that the partition generated by  $C$  is finer than the partition generated by  $D$ , Let us notice that the concept of dependency discussed above corresponds to that considered in relational databases. Summing up:  $D$  is totally (partially) dependent on  $C$ , if employing  $C$  all (possibly some) elements of the universe  $U$  may be uniquely classes to blocks of the partition  $U/D$ .

### 5.3.5 Achievements

Rough sets are used to select community detection algorithms based on their ability to form a good combination independent of their individual accuracy. The prediction made by a constituent classifier about the category of data instances is not considered to make the decision but the way a constituent classifier makes the partition on data set is the main consideration of the method. Usually, people deal with two kinds of partitions defined by any classifier: the partition defined on a given sample and the partition on the whole universe of objects (including unseen objects). The condition in the definition of  $f_{REC}$  is expressed using the partition on a sample but here it is assumed that the condition is preserved on the partition of the whole universe too.

## 5.4 Theoretical results

Let  $P_1 = \{P_{11}, P_{12}, \dots, P_{1k}\}, P_2 = \{P_{21}, P_{22}, \dots, P_{2k}\}, \dots, P_n = \{P_{n1}, P_{n2}, \dots, P_{nk}\}$  be the Partitions generated by community detection algorithms  $a_1, a_2, \dots, a_p$  on a given network  $G = (V, E)$ . Let  $SP = \{X_1, X_2, \dots, X_t\}$  be the super partition of  $P_1, P_2, \dots, P_n$ .

Now an optimal community structure is an another partition  $P^*$  of size  $k$ , such that  $P^* \subset SP$  and  $f(P^*)$  is optimum with respect to certain criteria.

First a quality measure is defined for the considered set of algorithms producing parti-

tions. Let us consider two partitions  $P, Q$  of  $U$  and an meta community extractor functions  $f : Q \rightarrow P$ .

The error of  $f$  relative to  $P$  is defined as:

$$ErP(f) = \sum_{x \in Q} \frac{|x|}{|U|} \left( 1 - \frac{|x \cap f(x)|}{|x|} \right) = \frac{1}{|U|} \sum_{x \in Q} (|x| - |x \cap f(x)|) \quad (5.2)$$

Optimality of  $f$  relative to  $P$  is defined as:

$$ErP(f) \leq ErP(g) \quad \forall \quad g : Q \rightarrow P \quad (5.3)$$

**Theorem 1:** Newly obtained meta-partition is an estimation to the optimal community structure of the network and converges towards optimal community structure with increasing number of input partitions.

**Proof:** Let  $u \in A$  and  $u$  corresponds  $x \in SP$  Then error of  $f$  corresponding to  $u$  is  $1 - \frac{|x \cap f(x)|}{|x|}$  To show that  $f$  is optimal, let  $g : SP \rightarrow P$  be any other function. Let  $x \in SP$  be arbitrary and  $f(x) = C_a$  and  $g(x) = C_b$ . By definition of  $f$   $|x \cap C_a| \geq |x \cap C_b|$ . Therefore  $\sum_{x \in SP} \frac{|x|}{|U|} (1 - \frac{|x \cap C_a|}{|x|}) \leq \sum_{x \in SP} \frac{|x|}{|U|} (1 - \frac{|x \cap C_b|}{|x|})$ . Therefore  $Error(f) \leq ErP(g)$  i.e.,  $f$  is optimal.

**Theorem 2:** Newly obtained meta-partition has better community structure than its contributing partitions.

**Proof:** Let  $P_r = \{P_{r1}, P_{r2}, \dots, P_{rk}\}$  be a partition corresponding to a input communities determined by a particular algorithm  $a_r$ . If  $a_r$  performs better than  $f$  then there exists a one to one correspondence of  $P_r$ , partition corresponding to algorithm  $a_r$ , and  $P$ , partition corresponding to the given categories. Let  $h : P_r \rightarrow P$  be this correspondence. Since  $SP$  is a refinement of  $P_r$ ,  $H : SP \rightarrow P$  can be defined such that, for any  $x \in SP$



$x$  is a proper subset of only one  $P_{ri}$  and  $H(x) = h(P_{ri})$ . Now  $Error(a_r)$  is same as  $Error(H)$ , (by definition of  $H$ ). But  $Error(H)$  can't be less than  $Error(f)$ . Therefore no constituent classifier perform better than  $f$ .

**Theorem 3:** Newly obtained meta-partition converges towards optimal community structure faster when input partitions have more mutual diversity.

**Proof:** The smaller the dependency, the greater the diversity of the input algorithms. A set of attributes  $P^*$  depends totally on a set of attributes  $C$ , denoted  $C \Rightarrow P^*$ , if all values of attributes from  $P^*$  are uniquely determined by values of attributes from  $C$ . In other words,  $P^*$  depends totally on  $C$ , if there exists a functional dependency between values of  $P^*$  and  $C$ . Formally dependency can be defined in the following way. Let  $P^*$  and  $C$  be subsets of  $A$ . We will say that  $P^*$  depends on  $C$  in a degree  $k$  ( $0 < k < 1$ ), denoted  $C \Rightarrow_k P^*$ , if  $k = \gamma(C, P^*) = |POS_C(P^*)|/|U|$ , where  $POS_C(P^*) = \cup_{X \in U/P^*} \underline{C}(X)$ , called a positive region of the partition  $U/P^*$  with respect to  $C$ , is the set of all elements of  $U$  that can be uniquely classified to blocks of the partition  $U/P^*$ , by means of  $C$ . Obviously

$$\gamma(C, P^*) = \sum_{X \in U/P^*} \frac{|\underline{C}(X)|}{|U|}. \quad (5.4)$$

If  $k = 1$  we say that  $P^*$  depends totally on  $C$ , and if  $k < 1$ , we say that  $P^*$  depends partially (in a degree  $k$ ) on  $C$ . The coefficient  $k$  expresses the ratio of all elements of the universe, which can be properly classified to blocks of the partition  $U/P^*$ , employing attributes  $C$  and will be called the degree of the dependency. It can be easily seen that if  $P^*$  depends totally on  $C$  then  $IND(C)$  subset  $IND(P^*)$ . This means that the partition generated by  $C$  is finer than the partition generated by  $P^*$ .

## 5.5 Experiments and results

In this experiment, we combine the communities of complex networks determined by several algorithms with Rough Sets theory to obtain a new optimal community structure of the network. We obtained improved results of community detection in less time by combining results of approximate algorithms and are shown in the tables 5.5.3 & 5.3.

### 5.5.1 Experimental Designs

We performed two different experiments to assess the performance of the proposed consensus community detection algorithm. First experiment is designed to evaluate the communities obtained by consensus method in terms of modularity and time. Second experiment is designed to explore the usefulness of consensus community detection method in terms of total gain in modularity and time compared to the exact algorithm, when consensus of several approximate version of same algorithm is considered. Experiments are conducted over several real networks 2.2 to compare the results of our algorithm in terms of modularity and time computed in seconds. The details of the several experiments and the analysis of the results are given in the following subsections.

### 5.5.2 Experimental setup to compare consensus community detection

In this experiment we tried to assess the usefulness of proposed consensus community detection method. For this purpose we have considered a group of community detection algorithms and used their outputs. Experimental steps are as follows:

**Algorithms:** The community detection algorithm proposed in sec. 3.5 is used in the

form of its three approximate variant, using LSH, using Nystrom sampling and using random sketches. the consensus community is constructed based on the output of above three variant.

**Network data:** Different types of real network data is taken, small, large, very sparse and relatively dense and they are discussed in sub-sec. 2.2.

**Results:** Compared the community structure obtained by consensus algorithms, metric based community detection algorithm and its three approximate versions. The results are shown in terms of modularity and execution time(seconds) in table 5.5.3.

**Observation:** It can be observed from table 5.5.3 that consensus community detection algorithm using approximate algorithms provides better modularity than the exact algorithm in less time.

### 5.5.3 Experiment on optimal community structure estimation

### 5.5.4 Results analysis and achievements

In this subsection, we have described the analysis of the results obtained in our experiments shown. The results obtained in the experiment justify that the consensus community extracted by proposed method is more useful for complex network compared to other community detection algorithms. The results obtained in the second experiment show that any community detection algorithm which reveals different community structure with different parameter setting can be improved by the consensus community method. However it is possible for some algorithm to output approximate results in sub-linear time, consensus community detection on a group of approximate algorithms can produce better than exact results in less time than the actual algorithm.

## 5.6 Conclusion

In this chapter, we present a comparative analysis of communities detected by several algorithms to redefine new community structure using rough set theoretic framework. We have created an information system of communities which are partitions on the set of nodes in a network. The outputs of different community detection algorithms constitutes super partition of the set of nodes. We described an algorithm for marging super partition until it constitutes desired number of communities. Applied to a diverse collection of large scale network datasets, the analysis reveals that (1) the different detection algorithms extract different structures with respective constrained; (2) the structure of communities that arise in our method is closest to that of communities of other algorithms in terms of mutual indecernibility; and (3) consensus communities extracted from heterogeneous community detection algorithms determine much better meta community of the same network in terms of modularity. The optimal combination of the communities obtained from several algorithms in-crease the modularity resultant community structure. This combination method is computationally very efficient when approximate algorithms are used.

Table 5.2: Experiment on optimal community structure estimation

Networks	Exact(M)	Exact(T)	LSH(M)	LSH(T)	Nystrom(M)	Nystrom(T)	Sketches(M)	Sketches(T)	RO(M)	RO(T)
Facebook	0.5472	6	0.5421	1	0.5405	1	0.5397	1	0.5526	2
Gplus	0.4056	661	0.4021	115	0.4010	92	0.4004	100	0.4061	77
Twitter	0.3709	398	0.3669	68	0.3656	54	0.3650	59	0.3763	45
Epinions1	0.1447	292	0.1437	56	0.1434	44	0.1432	48	0.1419	37
LiveJournal1	0.1458	969	0.1439	179	0.1433	143	0.1430	156	0.1466	119
Pokec	0.0295	901	0.0287	173	0.0285	138	0.0283	150	0.0302	115
Slashdot0811	0.0125	382	0.0117	74	0.0114	59	0.0113	64	0.0143	49
Slashdot0922	0.0168	389	0.0162	75	0.0160	60	0.0159	65	0.0182	50
Friendster	0.0536	1477	0.0525	280	0.0522	224	0.0520	243	0.0546	187
Orkut	0.0690	1074	0.0678	203	0.0674	163	0.0672	177	0.0697	136
Youtube	0.0936	578	0.0930	111	0.0928	89	0.0927	97	0.0913	74
DBLP	0.4215	341	0.4171	57	0.4157	46	0.4150	50	0.4286	38
Arxiv-AstroPh	0.5081	197	0.5035	33	0.5021	26	0.5013	28	0.5128	22
web-Stanford	0.3908	437	0.3866	74	0.3853	59	0.3846	64	0.3972	49
Amazon0601	0.2650	520	0.2621	93	0.2612	74	0.2607	81	0.2668	62
P2P-Gnutella31	0.0523	124	0.0523	24	0.0520	19	0.0520	21	0.0507	16
RoadNet-CA	0.1692	599	0.1664	107	0.1655	86	0.1651	93	0.1683	71
Wiki-Vote	0.2095	39	0.2083	7	0.2079	6	0.2077	6	0.2045	5

Table 5.3: Summary of results

Networks	Exact(M)	Exact(T)	RO(M)	RO(T)	Modularity Gain	Time Gain
Facebook	0.5472	6	0.5526	2	0.0054	5
Gplus	0.4056	661	0.4061	77	0.0005	584
Twitter	0.3709	398	0.3763	45	0.0054	353
Epinions1	0.1447	292	0.1419	37	-0.0028	255
LiveJournal1	0.1458	969	0.1466	119	0.0008	850
Pokec	0.0295	901	0.0302	115	0.0007	786
Slashdot0811	0.0125	382	0.0143	49	0.0018	333
Slashdot0922	0.0168	389	0.0182	50	0.0014	339
Friendster	0.0536	1477	0.0546	187	0.0010	1290
Orkut	0.069	1074	0.0697	136	0.0007	939
Youtube	0.0936	578	0.0913	74	-0.0023	504
DBLP	0.4215	341	0.4286	38	0.0071	303
Arxiv-AstroPh	0.5081	197	0.5128	22	0.0047	176
web-Stanford	0.3908	437	0.3972	49	0.0064	388
Amazon0601	0.265	520	0.2668	62	0.0018	458
P2P-Gnutella31	0.0523	124	0.0507	16	-0.0016	108
RoadNet-CA	0.1692	599	0.1683	71	-0.0009	528
Wiki-Vote	0.2095	39	0.2045	5	-0.0051	34

# Chapter 6

## Conclusion

In this work the fast and optimal algorithms for community detection on large network is described and analyzed. We demonstrated and analyzed a new approaches to network community detection via metric space induced by the complex network. The interesting problem of the nearest neighbor within the nodes of a complex networks are studied and applied for community detection. We have used geometric framework for network community detection instead of traditional graph theoretic approach or spectral methods. We presented the efficient computation of network community detection using low rank approximation. Our techniques can be applied for quick analysis of very large complex network. Finally the theoretical upper bound of network community detection algorithms is analyzed using the notion of data complexity and tried to estimate the optimal bound by heterogeneous combination of communities.

Complex network analysis is a large and growing body of research on the measurement and analysis of relational structure. Issues pertaining to data collection, analysis of single networks, network comparison, and analysis of individual-level covariates are discussed, and a number of suggestions are made for avoiding common pitfalls in the

application of network methods to substantive questions. The complex network field is an interdisciplinary research program which seeks to predict the structure of relationships among social and natural entities, as well as the impact of said structure on other natural phenomena.

The objective of this thesis is to bridge the gap between two important research directions, 1) complex network analysis, which deals with large real graphs but generally studied via graph theoretic analysis or spectral analysis and 2) data analysis in metric space using geometric distances, nearest neighbor search, which is a fundamental computational tool for large data analysis and low-rank approximation of distance matrix. Network community detection is not easy NP-Hard like data clustering due to the lack of good heuristics. Both, Graph traversal based methods and spectral methods are computationally overloaded due to the verification of objective function value, required to guide next iteration. Rich literature of clustering are not suitable for graph data. These observations motivates us for a transformation from complex network to Metric Space. But the metrics developed so far on graph (like Shortest path, Jaccard similarity and Euclidean distance between adjacency vectors) are less successful for network community detection in terms of conductance and modularity so there is a need for development of a good metric which works better on complex network. Aim of this work is the development of fast and accurate algorithms for community detection of large network. Competitiveness of the several approximation methods being analyzed with respect to their modularity value and computational complexity. The main contribution of the thesis is explained via the following chapters: 1) Chapter 2: Network community detection on metric space 2) Chapter 3: Nearest Neighbor search in Complex Network for Community Detection, 3) Chapter 4: Low rank approximations for community detection of very large networks, and 4) Chapter 5: Optimal evaluation of network community.



We develop the notion of a metric among the nodes using some new matrices derived from the modified adjacency matrix of the graph. The developed metric space is flexible over the networks and can be tuned to enhance its community structure. We also proposed the community detection algorithms on induced metric space and analyze the results and complexities of the developed algorithms.

The main achievement of the work was to use the rich literature of clustering in metric space. Clustering is easy NP-Hard in metric space, whereas network community detection is NP-Hard. The results obtained with our approach were very competitive with most of the well known algorithms in the literature and justified over the large collection of datasets. Our algorithm converges automatically to optimal clustering. It doesn't require verifying objective function value to guide next iteration, like popular approaches, thus saving the time of computation.

The main contributions of this work are, 1) development of the concept of nearness between the nodes of a complex network, 2) comparing the proposed nearness with other notions of similarities, 3) study and experiment on approximate nearest neighbor search for complex network using M-tree and LSH, 4) design of efficient community detection algorithm using nearest neighbor search. We observed that nearest neighbor between network nodes is a very efficient tool to explore better community structure of the real networks. Further several efficient approximation scheme are very useful for large networks, which hardly made any degradation of results, whereas saves lot of computational times.

The results obtained in the first experiment justify that the proposed distance is more useful for complex network to extract the community structure compared to other measures of similarity. The results obtained in the second experiment justify that the proposed algorithm for community detection is very efficient compared to other existing

methods in terms of modularity and time. The results obtained with our approach are very competitive with most of the well known algorithms in the literature and this is justified over the large collection of datasets. On the other hand, it can be observed that time (second) taken by our algorithm is quite less compared to other methods and justify the theoretical findings.

In this chapter we have proposed an algorithm to reduce the dimensionality of the datasets such that after applying partitioning on reduced network we get almost same results as with the original network via Nystrom sampling and random sketches. For evaluation of the proposed approaches on complex network we applied it in community detection problem. The results obtained using our methods are very competitive with most of the well known algorithms exists in the literature and this is verified on collection of real networks. On the other-hand, it can be observed that time taken by our algorithm is quite less compared to popular methods. In this work we will explore the two low rank approximation methods, Nystrom sampling and random sketching and they are described below. Low-rank methods are very fast to compute communities from massive networks by preserving the approximated results with moderate lower bound

In this chapter, we compare communities of complex networks determined by several algorithms using Rough Sets theory and tried to combine them to obtain an optimally community structure of the network. A community detection algorithm determines a partition of the network. Given several partitions of same network obtained from the output of different community detection algorithms, we tried to generate a new partition of same size with better modularity than all. The comparison of partitions are done using Rough Sets by assigning the minimum discernibility of the new partition with all others. The optimal combination of the communities obtained from several algorithms increase the modularity resultant community structure. This combination method is

computationally very efficient when approximate algorithms are used.

# Publications

## List of papers of the author:

### Journals:

1. Suman Saha, S.P. Ghrrera, *Nearest Neighbor search in the Metric Space of Complex Network for Community Detection*, Information, 8 (Special Issue: Online Social Networks and Implications), 2016. [**Scopus**]
2. Suman Saha, S.P. Ghrrera, *Network Community Detection on Metric Space*, Algorithms, 8 (Special Issue: Clustering Algorithms), 680-696, 2015. [**Scopus, Web of Science**]
3. Mohit Sharma, Suman Saha, *An Efficient Round robin Minimum Spanning Tree Algorithm in MapReduce Framework*, INROADS-An International Journal of Jaipur National University 3 (1s), 106-110, 2014.
4. C.P. Singh, Suman Saha, S.K. Saurabh, *Link Analysis to Visualize a Web Graph*, International Journal of Computer Science Issues (IJCSI), Vol. 9, Issue 3, p247, 2012.

**Conference:**

1. A Rawat, S Saha, SP Ghrera, *Distributed and Time Efficient Ranking System*, ICIIP 2015. [**Scopus**]
  2. LS Patel, S Saha, SP Ghrera, *Efficient Nystrom method for Low Rank Approximation and Error Analysis*, ICIIP 2015. [**Scopus**]
  3. M Sharma, S Saha, *Graph based approach for minimum multicollinearity highly accurate regression model explaining maximum variability*, Confluence 2014, 5th International Conference.
  4. K Arvind, J Suchi, S Suman, *Reduction in Searching Time of Inverted Index using Bloom Filter*, ERCICA-2014. [**Scopus**]
  5. M Sharma, S Suman, *Robust Best Regressor Model with Minimum Multicollinearity*, I2CT-2014, Pune, India.
  6. Parul Gupta, Suman Saha, *Approximate data mining using sketches for massive data*, CIMTA-2013, Procedia Technology, Elsevier. [**Scopus**]
-