AN ADAPTIVE ANALYZER FOR LARGE SOCIAL NETWORKS USING DISTRIBUTED CRAWLING

A THESIS

submitted in partial fulfilment of the requirement of the degree of

DOCTOR OF PHILOSOPHY

to

J.C. BOSE UNIVERSITY OF SCIENCE & TECHNOLOGY, YMCA

by

ATUL SRIVASTAVA

Registration No: Ph.D.-01-2K12

Under Supervision of

Dr. ANURADHA
ASSISTANT PROFESSOR
J.C. Bose University of Science & Technology, YMCA, Faridabad

Dr. DIMPLE JUNEJA
FORMER DEAN (R & D)
Poornima University, Jaipur

Department of Computer Engineering

Faculty of Engineering and Technology

J.C. Bose University of Science & Technology, YMCA, Faridabad

Sector-6, Mathura Road, Faridabad, Haryana, India

July 2019

DEDICATION

Dedicated to my wife and daughter…

DECLARATION

I hereby declare that this thesis entitled **"AN ADAPTIVE ANALYZER FOR LARGE SOCIAL NETWORKS USING DISTRIBUTED CRAWLING"** by **ATUL SRIVASTAVA**, being submitted in fulfilment of requirement for the award of Degree of Doctor of Philosophy in the Department of Computer Engineering under Faculty of Engineering and Technology of J.C. Bose University of Science and Technology, YMCA, Faridabad, during the academic year March 2013 to July 2019, is a bonafide record of my original work carried out under the guidance and supervision of **Dr. Anuradha, Assistant Professor, Department of Computer Engineering, J.C. Bose University of Science and Technology, YMCA, Faridabad** and **Dr. Dimple Juneja, Former Dean, Research and Development, Poornima University, Jaipur** has not been presented elsewhere.

I further declare that the thesis does not contain any part of any work which has been submitted for the award of any degree either in this university or in any other university.

ATUL SRIVASTAVA

Ph.D-01-2K12

CERTIFICATE

This is to certify that this thesis entitled **“AN ADAPTIVE ANALYZER FOR LARGE SOCIAL NETWORKS USING DISTRIBUTED CRAWLING”** by **ATUL SRIVASTAVA,** being submitted in fulfilment of the requirements for the Degree of Doctor of Philosophy in Department Of Computer Engineering, under faculty of Engineering and Technology of J.C. Bose, University of Science and Technology, YMCA, Faridabad, during the academic year March 2013 to July 2019, is a bonafide record of work carried out under our guidance and supervision.

We further declare that to the best of our knowledge, the thesis does not contain any part of any work which has been submitted for the award of any degree either in this university or in any other university.

More than two research papers in prestigious refereed journals have been published and all other academic requirements are fulfilled.

The thesis is based on the individual, original work of the candidate, which is previously unpublished research work and the thesis does not contain any material that infringes the copyright of any other individual or organization and does not hurt the sentiments of any individual(s) or religion(s). The information such as text, tables, equations, diagrams, figures, charts, and photographs taken from sources such as published work, like research papers, books, periodicals, web sites and other resources has been cited appropriately. Further, the opinions expressed or implied in the thesis are entirely of the candidate.

Dr. ANURADHA	**Dr. DIMPLE JUNEJA**
ASSISTANT PROFESSOR	Former DEAN (Research & Development)
J.C. Bose University of Science & Technology, YMCA, Faridabad	Poornima University, Jaipur

ACKNOWLEDGEMENT

I would first like to thank my supervisors Dr. Anuradha, Assistant Professor, Department of Computer Engineering, J.C. Bose University of Science and Technology, YMCA, Faridabad and Dr. Dimple Juneja, Former Dean, Research and Development, Poornima University, Jaipur for their mentoring and guidance during this research work. They have not only inspired, guided and supported but also made my dream of pursuing and completing the research work true. I am grateful to both of them for providing a cultivating environment that ropes liberty, open consultations, and constructive criticism, without which this work would not have been possible. They have always encouraged me to explore innovative things and were always there to guide me for improvisation by giving their invaluable suggestions. They have taught me to be strong and push forward when it is hard. I thank both of them from all my heart for the help in all aspects of my Ph.D. I am indebted to the long 'wisdom talks', which will benefit me throughout my future. I would like to express my sincere gratitude to Prof. Komal Bhatia, Prof. Atul Mishra and Prof. C. K. Nagpal, Professors, Department of Computer Engineering, J.C. Bose University of Science and Technology, YMCA, Faridabad, for trusting and believing in me. All of them inspired me to set the research standards high and not to compromise with them. They have cheered for me during my ups and cheered me up during my downs. I am extremely honoured to have worked and interacted with both of them. I am also thankful to all my students, faculty and staff members of the department who helped me directly or indirectly in completing my work. I would like to thank my wife, Dr. Shalini Srivastava, for inspiring me and showing me that there are no limits and I can pursue whatever I am passionate about. She is the most amazing person I have met and I am very blessed to have her near me. She constantly cheered me when I was down and helped me when I needed advice and aid on my research.

Finally, I would like to thank my supportive and affectionate family for their constant support. I am grateful to my father and mother for a wonderful education, my brothers for supporting my education.

(Atul Srivastava)

ABSTRACT

Social networks have gained overwhelming popularity among every sect of the society after inception of online social networks. Social networks provide easy and interactive platform for people to share their interests, ideas, like, dislikes, orientations etc. The association among entities on social networks is formed in terms of interaction among them. The interactions are real time and therefore the exact behavior of individuals is reflected. The study of this real-time behavior enables to draw analytical conclusions. Social Network Analysis (SNA) is a collection of procedures, tools, parameters, techniques and softwares to carry out this analytical exercise on social network data. Due to availability of real time behavioural data and promising results, SNA finds significant applications in various domains like communication studies, economics, information science, terrorism prevention, organizational studies, anthropology, social psychology, biology etc.

Despite wide range of applications, SNA faces prominent challenges. The very first challenge for SNA is data collection from social networks. The size of social networks is gigantic and to acquire such huge amount of data requires infeasible amount of time and computing resources. Moreover, if such huge amount of data is gathered on local storage media, still there is a requirement of same or more amounts of time and computing resources to analyze it. A logical solution to this problem is to collect sample of the social network instead of collecting complete network. The size of sample is fairly smaller than the size of original network yet the structural properties of the original network should be exactly reflected in sample. BFS, MHRW, FS, NBRW-rw are some of the significant algorithms for sampling social network. The sampling algorithms of social networks face major challenge of *'biasing'*. Biasing leads social data collector to get attracted towards one kind of nodes (generally higher degree nodes). This degrades quality of sample significantly. This challenge has been addressed in the proposed framework CCSC. CCSC is an adaptive framework for crawling social networks which does not get biased and is able to capture exact clustered view of the original network in collected sample.

Another major challenge is dynamic behaviour of social networks. Individuals on social networks tend to continuously change their associations. The offline sample stored in local media may soon get obsolete. A framework FPBC has been proposed to verify and maintain freshness of the social network dataset. FPBC is an extension of CCSC.

The sample of social network is much smaller than the original social network, yet its size is big enough to create significant time constraints for analytical techniques like community detection. Analysis of social networks is identifying structural characteristics of the social networks and making conclusions based on those characteristics. The characteristics like degree distribution, centrality measurements, modularity, cliques, communities etc. are prominently preferred for SNA. The most widely used SNA technique is detection of communities in the social network. The community is a group of nodes which are more closely connected to each other as compared to their connections to rest of the nodes of the network. Community detection is a time consuming process. Several significant algorithms exist in the literature. The objective of community detection technique is to detect high quality communities in minimum possible time. Two approaches are majorly dominant in community detection; one which treats communities as standalone groups and second which considered communities to be overlapping. Maximal clique has significant application in detection of overlapping community structure in the network. In this work, a time efficient algorithm to find maximum clique, MCF, has been proposed which uses four pruning steps to achieve competent results and beats prominent similar algorithms in time. MCF is used in proposed community detection technique, KCUF. KCUF is a k-clique based time efficient community detection technique. It produces correct results in less time as compared to other similar algorithms. KCUF optimizes time at identification of cliques using MCF and at processing of maximal cliques using Union Find data structure.

All the frameworks are tested on standard data sets and social networks. The results are competent.

TABLE OF CONTENTS

LIST OF PROPOSED ALGORITHMS

LIST OF TABLES

LIST OF FIGURES/GRAPHS

LIST OF ABBREVIATIONS

ABBREVIATION	DESCRIPTION
SNA	Social Network Analysis
OSN	Online Social Network
DSN	Dynamic Social Network
MEDLINE	Medical Literature Analysis and Retrieval System Online
IMDB	Internet Movie Database
URL	Uniform Resource Locator
IP	Internet Protocol
CCSC	Cluster Coverage Sampling Crawler
FPBC	Foot Prints Based Crawler
TINB	Topical Interaction Network Builder
MCF	Maximum Clique Finder
KCUF	K-Clique Based Community Detetection using Union Find
DyNet	Dynamic Neural Network Toolkit
NETTEST	Estimating a Terrorist Network’s Structure
HITS	Hyperlink-Induced Topic Search
WWW	World Wide Web
BFS	Breadth First Sampling
NBRW-rw	Non-Backtracking Random Walk with reweighting
FS	Frontier Sampling
RW	Random Walk
MHRW	Metropolis Hasting Random Walk
USDSG	Unbiased Sampling in Directed Social Graph
MCMC	Markov Chain Monte Carlo
HTTP	Hyper Text Transfer Protocol
SFC	Semantic Focused Crawler
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
CPM	Clique Percolation Method

SPM	Sequential Clique Percolation
ECL	Extended Clique Percolation
FCA	Formal Concept Analysis
DIMACS	Centre for Discrete Mathematics and Theoretical Computer Science
EXP	Expression
Cond_Table	Condition Table
Op_List	Operators List
HTML	Hyper Text Markup Language
API	Application Programming Interface
NMSE	Normalized Mean Squared Error
ODP	Open Directory Project
RDF	Resource Description Framework

CHAPTER 1

INTRODUCTION TO ASPECTS OF SOCIAL NETWORK ANALYSIS

1.1 MOTIVATION

Social networks have gained tremendous popularity in recent past due to advancements in computer science and internet services. Almost every individual is on some sort of social network. Social Network is a pact of individuals connected together through social and interactive relations to share interests, ideas, thoughts, likes/dislikes etc. The objective of social networks is to provide platform for interaction among individuals. Therefore, social networks contain real time social information and most realistic behaviour of the society or the domain to which the social network is related.

Due to the reflection of exact behaviour and orientation of individuals on social networks, several significant analytical conclusions can be drawn by capturing and analyzing the social network structure and information flow on social networks. This leads to an influential branch of study, research and application called Social Network Analysis (SNA) [9, 12]. SNA finds significant applications [53 - 56, 59, 60, 62 159, 160, 161] in various domains like communication studies, economics, information science, organizational studies, anthropology, social psychology, biology etc. Primary objective of SNA is to identify the conclusive outcomes of the structural and behavioural aspects of the social network. SNA includes procedures, tools, software, parameters, techniques etc. for social network data collection and analysis. However; carrying out SNA is not that easy.

The challenges for SNA are gigantic size of the social networks. Billions of users are there on online social networks like Facebook and Twitter. Moreover, collection of complete social network data is infeasible and therefore conventional researches are being done on sample of the social network. Ample number of sampling techniques are listed in literature[63, 67, 68, 69, 70, 71, 72, 75, 77]. In fact, sampling of social networks again comes with a challenge of biasing [63]. The quality of sample depends on its resemblance with the original network. An unbiased sampling technique is required which captures every structural characteristic of the social network in sample. Still the sample is also large and complex enough to make it difficult for analysing it. Thus a time efficient accurate analysis technique for social networks is very much apparent.

Social networks are dynamic in nature i.e. the information streaming on such networks continues to change very often. In addition, the individuals of the social network also tend to change their interconnections with time. Thus, a sample of a dynamic social network soon gets obsolete. Researchers have handled dynamism of social networks at analysis level [122 - 127]. Thus an approach to detect the deviation between sample and the original network is obligatory to keep the dataset fresh.

The wide range of applications of SNA clearly shows its promising outcomes. There are domain which are not inherently social networks but have interaction among their entities. If a social network like interaction network of the entities based on their interactions among themselves is created, SNA techniques can be directly applied to find analytical results from these domains also.

Owing to the above stated barriers in the success of SNA and its usefulness, a solution addressing the above issues is very much apparent. Thus, the motivation of carrying out this research work.

Upcoming sections briefly throw light on Social Network and its Analysis. A section is exclusively being devoted to research challenges prevailing in SNA on the basis of which various research objectives are being identified.

1.2 SOCIAL NETWORK

Social network is interconnection of socially active individuals which are generally called actors. Social networks provide a platform where people can connect with one another and share ideas, thoughts, opinions, likes and dislikes. People are offered to maintain societal relationships among themselves on social networks. Social networks are represented as graphs for computation purposes. The user/actors are denoted by nodes and the associations among themselves are denoted by edges.

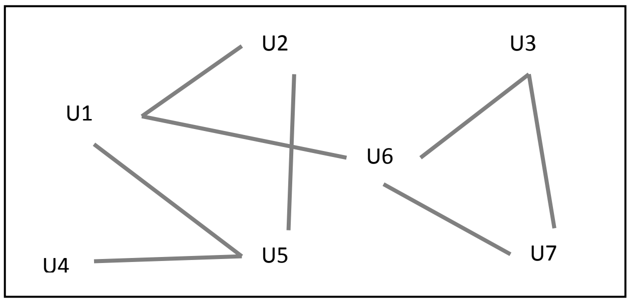


Figure 1.1 Graphical Representation of Social Network

Figure 1.1 represents an example of social network represented as graph. As the association among users of social network can be bidirectional (e.g. friends on facebook) or can be unidirectional (e.g. follow on twitter), the social graph can have directed of undirected edges. The association between two users implies that these users have already shared some information between them.

Table 1.1 Adjacency Matrix Representation of Social Network

Actors/ Users	**Actors/Users**						
	U1	***U2***	***U3***	***U4***	***U5***	***U6***	***U7***
U1	--	1	0	0	1	1	0
U2	1	--	0	0	1	0	0
U3	0	0	--	0	0	1	1
U4	0	0	0	--	1	0	0
U5	1	1	0	1	--	0	0
U6	1	0	1	0	0	--	1
U7	0	0	1	0	0	1	--

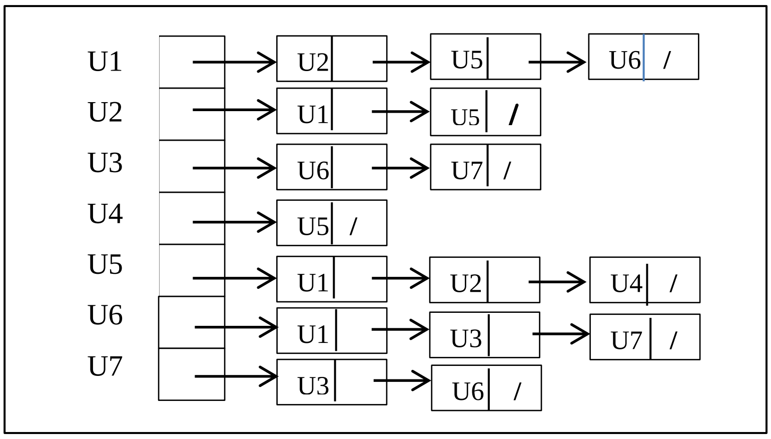


Figure 1.2 Example of Adjacency List

In fact, graphical representation is most suitable for understanding the structural properties of the social network but it is not convenient for machines. Hence, adjacency matrix or adjacency list is used an alternative representation. Table 1.1 represents social graph (shown in figure 1.1) as adjacency matrix. Figure 1.2 illustrates adjacency list for representation of graphs.

1.2.1 Online Social Networks

Remarkable growth in computation power of the machines and popularity of internet has brought social networks online. Online social networks are easy to access and users can comfortably connect with others and share ideas, interests, opinions etc. [1]. Social networking sites are web based services which enable users to create a public profile and connect with other people. The individuals are also privileged to manipulate the list of their friends and can modify privacy settings for visibility of the shared content. Users are also able to visit profiles of other connected users and see the information shared by them. Usually OSNs consist of three entities namely actors (users), links and groups. Users can be individuals or an organization etc. User need to register on the social networking sites. Few networking sites[1] allow users to browse the content without signing in, but to have full access users ought to sign in first. The users need to register with a pseudonym.

Social networks have emerged as prominent platform to analyze, represent, identify and estimate the correlations among individuals as well as among any kind of entities such as web pages, words, organizations, cells, people, animals, computers or devices or knowledge etc. [5, 6, 7, 10, 11]. At root level some of these domains are not inherently social networks but entities in these domains have connection between them. Such connection can be based upon the interaction between the entities. Therefore, social network prepared from the entities of nonsocial domains is called *Interaction Network* [174]. Next section discusses various aspects and significant benefits of interaction networks.

1.2.2 Interaction Network

Interaction network is a network of entities which can be connected by the fact, that they have some information common. For an instance consider a social network, in

[1] https://www.blogosphere.biz/

which two users are considered having connection between them if they are friends to each other or one of them is following the other. Users on social networks make connections if they have some common attributes such as common work place, similar political view, same study place, common interests in sports or in music etc. Social graphs obtained from prominent social networks are used for many social studies such as terrorism detection, monitoring epidemics, exit polls etc. [12].

While social networks are trivial interaction networks, interaction networks can be constructed with similar phenomenon for nontrivial domains also. For example, there are many websites which can be put in one category on the basis of some criteria, such as the content they are showing is similar, the technology on which the websites are developed etc. Alike social networks where social graphs are used to represent a social network, interaction networks are represented with web graphs. In web graphs, web pages are considered as vertices and if two webpages have common information (showing similar content/ using same technology), either an edge (undirected) can be put between them or if one page contains hyperlinks to other pages then directed edges can be put among them. This interaction network of web pages can be used for many purposes such as clustering the web pages and identifying a particular category of websites such as pornographic websites or websites spreading fanatic ideas or the recent technologies being used by the sites.

Whether it is a simple social network or an interaction network, all such networks offer real time data which can be further analyzed for lots of business benefits. Hence, an attempt is being made to throw light on SNA in the next section.

1.3 SOCIAL NETWORK ANALYSIS

SNA is not just a suggestive metaphor any more but it has developed into a well-equipped analytical paradigm. The strength it has got has come from theories, SNA tools, software, methods and researchers. The analysis can be focused upon the whole or on a particular part, on the structural relations or on behavior or attitude. The researchers either tend to analyse complete networks in which all the ties have specific relations or they can study the personal networks generally known as egocentric networks. The difference between both lies in the way of data collection. Personal or ego centric analysis can be carried out if the identities of egos are known.

Most of the social networks are egocentric. SNA is primarily based on several attributes some of which are defined as characteristics of graphs and some as distribution of information and behaviour of actors.

Fundamentally, SNA focuses on understanding the information contained in the network to study the structural information of social network. Few of the prominent definitions of SNA can be as follows:

(a) The mapping and measuring of relationships and flows between people [19].

(b) The technique focusing on uncovered patterns of people's interaction [20].

(c) A set of methods for the investigation of relational aspects of social structures [21].

1.4 ASPECTS OF SNA

As reflected in figure 1.3, SNA is a twin-step process i.e. data collection and exploring structural aspects of the social network from the data collected.

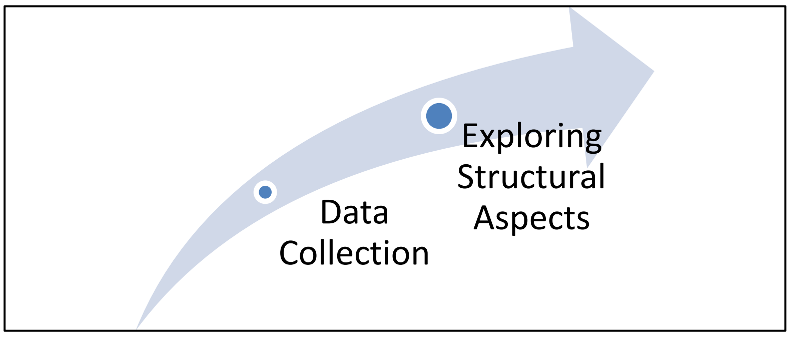


Figure 1.3: Social Network Analysis

Upcoming section briefly describes data collection from social networks.

1.4.1 Social Network Data Collection (Social Web Crawling)

In theory social data collection is mainly done in two ways – through elicitation and through registration [14]. Elicitation simply deals with questionnaire/survey while registration deals with extraction of information from registered entities/information. Later, due to rapid advancements in computer science, automated data collection became popular. There are many large databases which maintain huge datasets corresponding to specific fields. For instance, MEDLINE is the database of biomedical research papers and contains around two million records from 1995 to 1997 [18]. IMDB and LiveJournal are other such databases.

After advent of internet and then online social networks, real social network data collection has transformed significantly from being a lengthy rigorous process to an automated program driven data collection. A computer program enters into internet and collects data autonomously is called crawler. Conventional web crawlers are generally used to collect data for indexing purposes which is generally used by search engines, whereas social web crawlers focus on collecting the structural properties of the social network. Social web crawler starts with a particular user which is seed node for the crawler and later explores its friend list and finds more nodes to be crawled further. SNA techniques are applied on the social graph to dig the structural characteristics of the network defined by various components and is being briefly discussed in upcoming section.

1.4.2 Analysis of Social Graph

SNA majorly revolves around identifying the behaviours of the individuals in the network and behaviour of the network as whole. These are defined in terms of various characteristics of the social graph. SNA focuses on the characteristics of social graphs such as degree distribution in the graph [22], geodesic distance and diameter [23] of the graph, cliques and subgroups [23] in the network, maximum flow among the nodes [23], centrality of nodes [25], power of nodes in the network [25], small world effect in the network [22, 26], clustering coefficient of nodes in the graph [22, 26] and communities [27] in the graph. Community detection is most effective and widely used SNA technique. Since, every characteristic of the social graph conveys an analytical perception about the graph, hence, analysis a social graph inherently is equipped with many challenges. Few of these challenges are being presented in next section.

1.5 CHALLENGES IN SNA

On investigating the literature, it was discovered that there exist various challenges in SNA. However, few dominating ones are listed as follows:

1.5.1 Size of Social Network

The popular social networks are networks of acquaintance/communication [31], Live Journal [6], MySpace [32], Facebook [4], phone calls [33], collaboration [18, 34], sexual contact [35], paper citation [36], World Wide Web (www) pages network [37, 38], linguistic network [39] etc. All of these networks are gigantic in size acting as the biggest barrier in SNA. Elementarily, gigantic size of social networks puts following barriers:

(A) Long Processing Time

Huge amount of social activities is an obvious outcome of gigantic size of the social network. Approximately 1.3 billion monthly active users are on Facebook and 284 million monthly active users are there on Twitter [2, 3]. LiveJournal has more than 10 million actors in its acquaintance network. The size of the social network is continuously increasing. For social network analysis this gigantic amount of social network data has to be collected and then has to be processed. Even if computers have increased their computation power significantly in recent past but still the processing time with this amount data comes to be nonlinear.

(B) Large Computation Resource Requirement

A great amount of computing resources and storage media are required to handle such huge amount of social data. For instance, a 32 bit micro-processor is able to address 2^{32} bytes of the memory which limits the size of total physical memory to be 4 GB. If shortest paths in a graph having *n* vertices has to be computed, then the algorithm would take O(n^2) memory. If a node (actor/user) takes 4 bytes then 4 GB of memory can accommodate only $\sqrt{2^{32}}/_4 \approx$ 16,000 actors. This number is much smaller than the number of actors in a normal social network.

1.5.2 Graph Dynamism

Dynamism of social networks plays an impactful role in SNA. Most social networks, such as blogs, social media, co-authorship networks etc. are prone to evolve gradually due to continuous changes in the interaction patterns within them and frequently changing activities of actors [12].

Dynamic behavior of social networks directly influences the validity of conclusions drawn by social network analysis. The inferences derived hold only if the evolution pattern of the dynamic social network has been determined and suitable corrective modifications are carried out on the results. The user interactions are highly volatile in the network and users tend to form new connection while they may leave the old ones.

1.5.3 Diverse Representation of Social Networks

At abstract level the social networks are similar but at structural and implementation level the social networks are diverse. During development of SNA tools and software, such structural differences create a barrier because the tool developed for a particular social network may not work for a different social network. On the other hand, if the SNA framework has to be universal then it must include structural aspects of every social network and well equipped customization ability.

1.5.4 Biasing

To deal with gigantic size of social network, crawlers prefer to collect a sample of social network instead of crawling complete network. Sampling has an added challenge in it which is called 'biasing' [40]. The crawlers may get biased towards similar kind of nodes. If the sample contains all similar nodes then then it is not a good representative of the social network.

On having a critical look at the literature and hence the identified challenges, following research objectives are being identified to be achieved during the course of this research study.

1.6 OBJECTIVES OF RESEARCH WORK

On the basis of literature studied in detail during initial phase of this research work, following objectives have been identified in the light of prominent applications of social network analysis and are achieved in this research work:

1. To design a social web data collector and freshness maintainer:

1.1 Collecting data from trivial social networks

1.2 Maintain freshness of the dataset of dynamic social networks

1.3 Collecting data from non-social web domains and give it structure of social networks

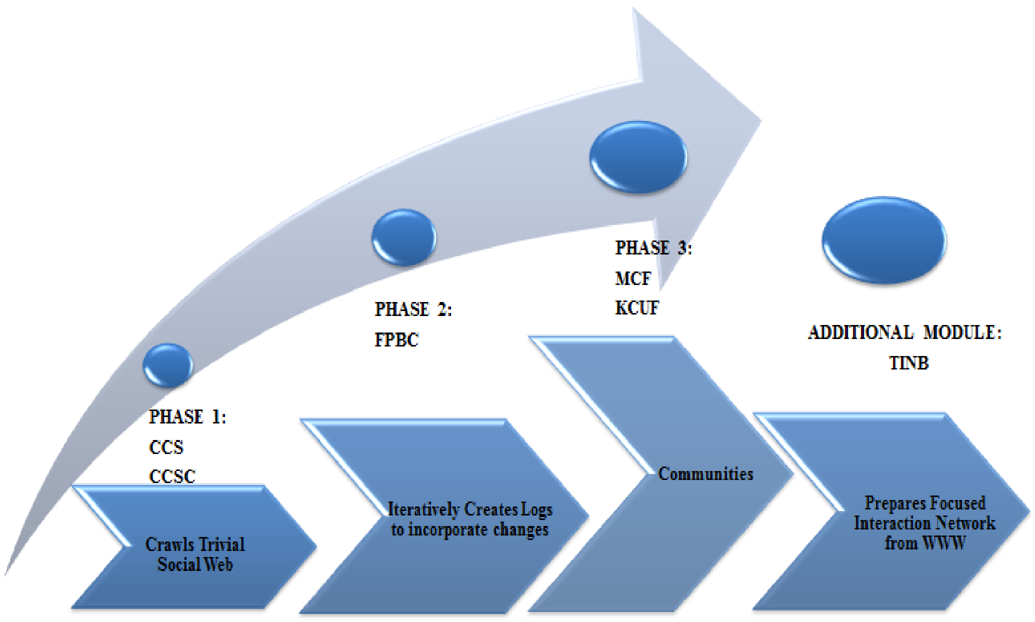
2. To develop an algorithm for community detection in the network

2.1 Fast algorithm to find clique in the graph

2.2 Extending Maximum Clique Finder to detect communities in social network

1.7 THE PROPOSED WORK

To achieve the above stated objectives, this research uniquely contributes a three-phased framework for social network data collection and analysis. The view of modules of the proposed framework is shown in figure 1.4.



PHASE 1:
CCS
CCSC
Crawls Trivial Social Web
PHASE 2:
FPBC
Iteratively Creates Logs to incorporate changes
PHASE 3:
MCF
KCUF
Communities
ADDITIONAL MODULE:
TINB
Prepares Focused Interaction Network from WWW

Figure 1.4 Modules in Proposed Framework

- The work began by investigating the literature and in support of the literature review, publications titled "*Social Network Analysis: Hardly Easy*" and *"A Walk Through Social Network Analysis: Opportunities, Limitations and Threats",* contributes to objective 1.

- *In order to achieve the rest of the* objectives the work proposes a three phased solution. Phase 1 and Phase 2 deals with data collection from social networks and maintaining the freshness of the sample collected for dynamic social networks. Phase 3 proposes a novel and time efficient *K-Clique Based Community Detection using Union Find* (KCUF) [44] algorithm which deploys *Maximum Clique Finder* (MCF) [45] algorithm which in turn finds maximum clique.
- Phase 1 proposes a crawler named as *Cluster Coverage Sampling Crawler*, henceforth would be referred as CCSC [41]. CCSC considers clustered structure of the social network and offers a solution to biasing and is a natural representative of the already clustered social network.
- In phase 2, CCSC is extended to become *Foot Prints Based Crawler* (FPBC) [42] for social network, which handles dynamic behavior of the social network. FPBC leaves its foot prints in a log file while crawling the social web. These foot prints are re-traversed by the proposed crawler to ensure freshness of the sample collected in previous crawl.
- The proposed framework also contributes a module termed as *Interaction Network Maker*, which is responsible for creating a social network from WWW which are not inherently social networks but can be expressed as social networks. The *Topical Interaction Network Builder* (TINB) [43] has been implemented in this module that successfully creates a topical social network of web pages from WWW. This topical social network can be treated in the same fashion as any other trivial social network.
- KCUF is a community detection framework based on k-cliques. It also finds maximum clique in any graph using MCF algorithm. Finding maximum clique is a NP-Hard problem [24] problem. MCF uses several optimization pruning strategies to make clique finding time efficient. KCUF is responsible for detecting communities in the social graph. Community detection is a significant aspect of social network analysis and is widely used in various fields of research. KCUF uses MCF for finding cliques and makes use of Union Find to merge similar communities into one.

1.8 STRUCTURE OF THE THESIS

The research work is principally covered in eight chapters as listed below:

Chapter 1 discusses the motivation for the research work and presents a brief idea of background concepts necessary for commencing the research work.

Chapter 2 illustrates the basics about social network and social network analysis. It briefs about the structural properties of social network and their significance in various applications

Chapter 3 discusses significant work done by researchers in various domains of social network analysis. The literature study has been classified according to the objectives. This chapter also presents the challenges of the research work, feasibility of research work and objectives to be achieved during the course of work.

Chapter 4 furnishes on Phase 1 and 2 of the proposed work i.e. data collector for trivial social networks and maintaining freshness of the dataset. Biasing is one of the most significant challenges in crawling social networks as discussed in the literature. A novel approach Cluster Coverage Sampling Crawler (CCSC) has been presented in this chapter which does not suffer from biasing. It also discusses Foot Prints Based Crawler (FPBC) which is extension of Cluster Coverage Sampling based crawler with the extended power of leaving foot prints in a log file. The foot prints of the crawler include the decider expression and a digest of the current sample. This log is used as a reference to test freshness of the dataset. It generates a sample of large social network like facebook with better resemblance with the actual network. It tries to cover every cluster in the social network be changing the decider expression for the next sampled node. It further furnishes on generating social network from non-social web domain www. This section presents a focused crawler for www which generates an interaction network of web pages.

Chapter 5 focuses on Phase 3 of the proposed work i.e. Analysis of Social Networks. This chapter starts with discussion of co-relation between finding clique in the graph and community detection. A fast algorithm to find maximum clique, Maximum Clique Finder, is presented which finds maximum clique in a large graph in feasible time.

Later in this chapter, K-clique based community detection algorithm KCUF has been discussed in detail which uses Maximum Clique Finder algorithm as sub routine.

Chapter 6 evaluates and analyses the proposed work on various performance metrics and claims the accuracy of proposed work.

Chapter 7 concludes the outcome of the work. It summarizes the major achievements of the research work and elucidates the scope for future work in this domain.

CHAPTER 2

THE BACKGROUND

2.1 INTRODUCTION

The chapter provides an insight into the background study concerning to social networks and SNA. It also presents various applications of SNA, issues flowing in SNA and various solutions that have been proposed so far for handling such issues. A section in the chapter is exclusively devoted to discussion about various metrics that are applied to measure the performance of a SNA tool. It is worth mentioning that these metrics would later be used to analyse the proposed work.

The upcoming section details about the social networks.

2.2 SOCIAL NETWORKS

As defined in previous chapter social network is collection of users and their interconnections. Every social network is similar in terms of basic terminology but in practice, different social network may differ at some fundamental aspects.

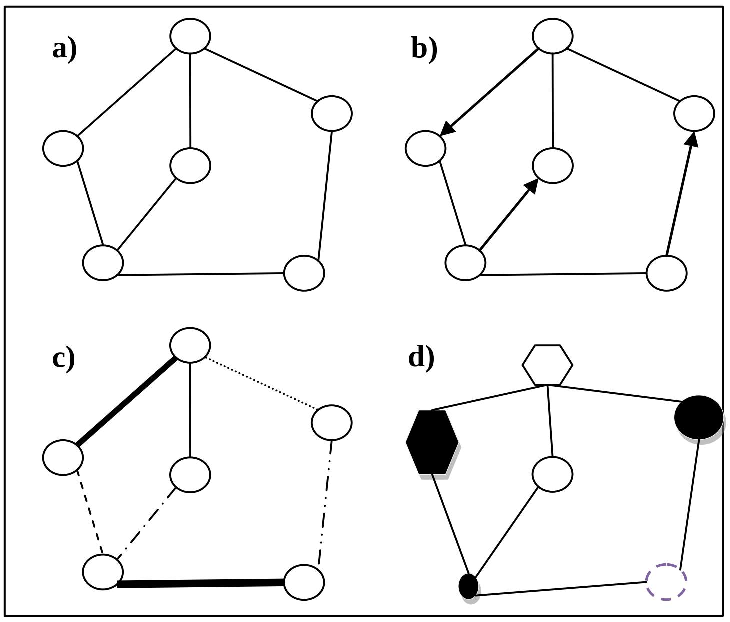


Figure 2.1 Varieties of Social Networks (a) Social Network With Homogeneous Associations Of Equal Weights and Homogeneous Nodes. (b) Social Network with Homogeneous Nodes and Heterogeneous Ties with Equal Weights. (c) Social

Network with Homogeneous Ties with Different Weights and Homogeneous Nodes. (d) Social Network Homogeneous Ties with Heterogeneous Nodes.

For instance a social network can have all homogeneous associations or heterogeneous associations among users. The users of the social network can be of same type or they can be of various categories. Figure 2.1 depicts few of the possibilities of social networks in terms of variety in associations and actors. The associations among actors of social network can be directed or undirected. The intensity of the association among users can be same for every connection or it can be different. For instance on Facebook the association is ‘friend’ which is undirected (for bidirectional relations) and has same weights for every association. But on a social network of authors working together, associations are undirected but have different weights depending upon the endeavors, two authors have had together. Similarly, social networks may contain homogeneous actors or may have heterogeneous actors. For instance, on Twitter, the users are not only people with twitter handle but the organizations also have their twitter handle, products have their twitter handle, which are different from individual twitter handles.

Social networks are prominent platforms to understand and analyse the interaction patterns of entities, their behavior and information propagation as Online Social Networks (OSNs). OSNs have grown significantly in last two decades [1]. Approximately 1.3 billion monthly active users are on Facebook and 284 million monthly active users are there on Twitter [2, 3]. Twitter is also called SMS of internet. The next section discusses inception of online social networks and their evolution.

2.2.1 Origin of Online Social Networks (OSNs)

Online social networks are popular among every sect of the society. The connections between users on OSNs are based upon their trust, common interests, political orientations, common habits etc. The data is easily available online for the analysis of relationships of people and their behavioral psychology. There has been an exponential growth from first online social network SixDgrees.com [1] (1997) to the largest social network of present time Facebook [4]. This is proven fact as approximately 1.3 billion monthly active users are on Facebook and 284 million monthly active users are there on Twitter [2, 3]. Figure 2.2 depicts a chronological order of inception of OSNs.

Users have discretion to share their data publically and may volunteer to share their personal information like birth place, study place or work place, date of birth etc. On some social networking sites such as Facebook [4], Orkut [7], LinkedIn [8] etc. the users need to ask for permission from other users to connect with them. In some other social networking sites like Flicker [5] and LiveJournal [6], registered users can connect with anyone without their consent.

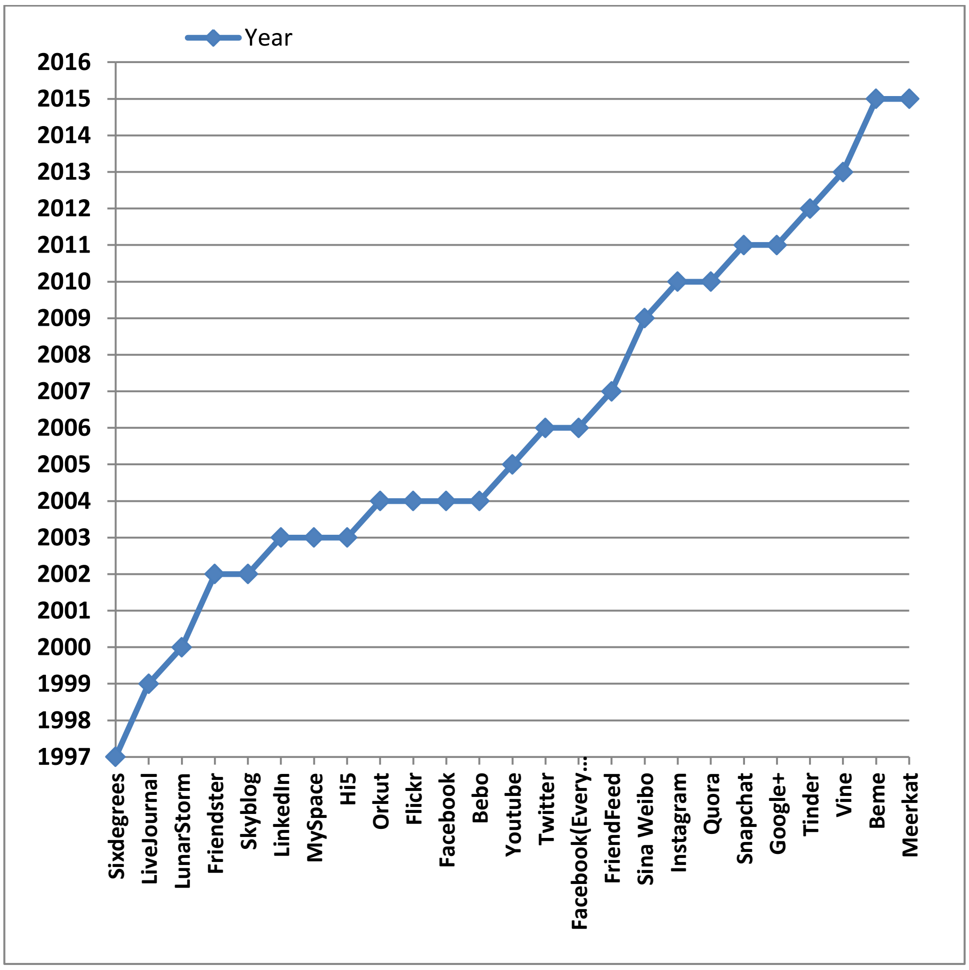


Figure 2.2 Chronological inceptions of Online Social Networks

Figure 2.3 depicts daily active users on some of significant online social networks. This data has been published by statista2 in 2018.

Figure 2.4 depicts various reasons of having people associated with each other.

On social networks the users are able to see associations of other users also and can visit to any user by traversing connections to connections. Users can see the personal information shared publically by other users. Few social sites like LinkedIn [8] have put a restriction on visiting other users by a particular user.

[2] www.statista.com

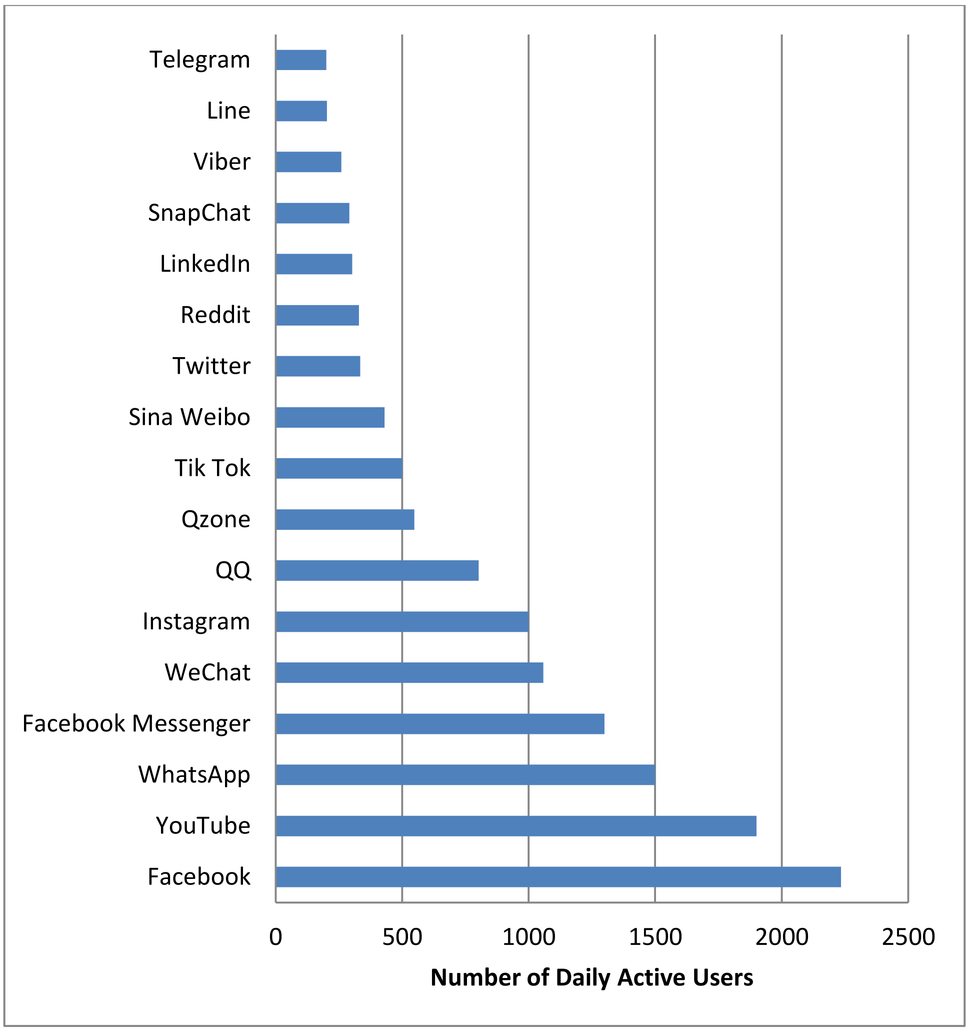


Figure 2.3 Active Users on Social Networks

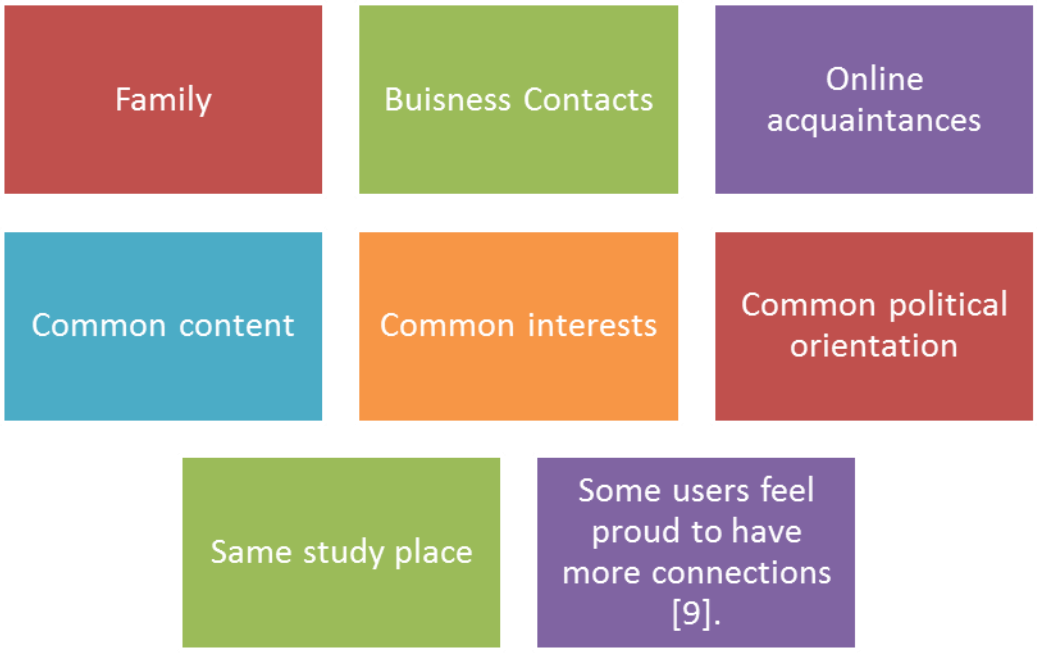


Figure 2.4 Bases for Interconnection among Users on Social Networks

The user can only visit a user which is two hops away from it. But most of the social networking sites, which are commonly targeted for social network analysis, do not put such restrictions. In order to understand the associations between users and the behavioral patterns, the data streaming on social networks is analyzed to generate business oriented results. At times, such results are also useful in deciding terrorism related activities, fake news, popularity of a person, topics of interest and so on. Before turning our attention to SNA, the upcoming section elaborates social web crawling which is done to collect the data thus streaming on a social network.

Every online social network or in general almost every social network is gigantic in size and dynamic in nature [12]. Dynamism of social networks plays an impactful role in Social Network Analysis. Most social networks, such as blogs, social media, co-authorship networks etc. are prone to evolve gradually due to continuous changes in the interaction patterns within them and frequently changing activities of actors [12].

Dynamic behavior of social networks directly influences the validity of conclusions drawn by social network analysis. The inferences derived hold only if the evolution patter of the dynamic social network has been determined and suitable corrective modifications are carried out on the results. The user interactions are highly volatile in the network and users tend to form new connection while they may leave the old ones. The community to which user belongs cannot be permanent. Evolution pattern of the network is proportional to migration extent of the users.

Further, the users on most of the social networking sites can create groups to have a similar content being shared by every user of the group. The groups can be open or restricted depending upon the discretion used by its owner.

Representing non-social domains as social networks seems promising due to prominent results obtained from SNA. Upcoming section elaborates on interaction network by an interaction network of computer/mobile games.

2.2.2 Interaction Networks of Games

The idea of interaction networks can be elaborated with following example. For instance, consider video games for computers and mobile phones. Generally computer games or mobile games are displayed in categories on the hosting web sites. The categories can be based on the *genre* of the games, or the *technology* used to develop the game, or the *features* of games etc. Information can be created for games based on categories. The games falling in same category can be associated with each other as they have certain attributes in common. An interaction network of games can be created where games based on similar *genre* or having same *features* can be connected. Figure 2.5 shows interaction network of mobile games prepared with respect to their *genre* from 9Apps website. Figure 2.6 elaborates a section of this interaction network where it is shown that the game named ‘Teen Patti’ is associated with two clusters, which means ‘Teen Patti’ fits into both categories. Each cluster represents a genre. By having prepared the interaction network, such analytical conclusions can be drawn easily. Figure 2.7 shows interaction network of computer games according to *genre*.

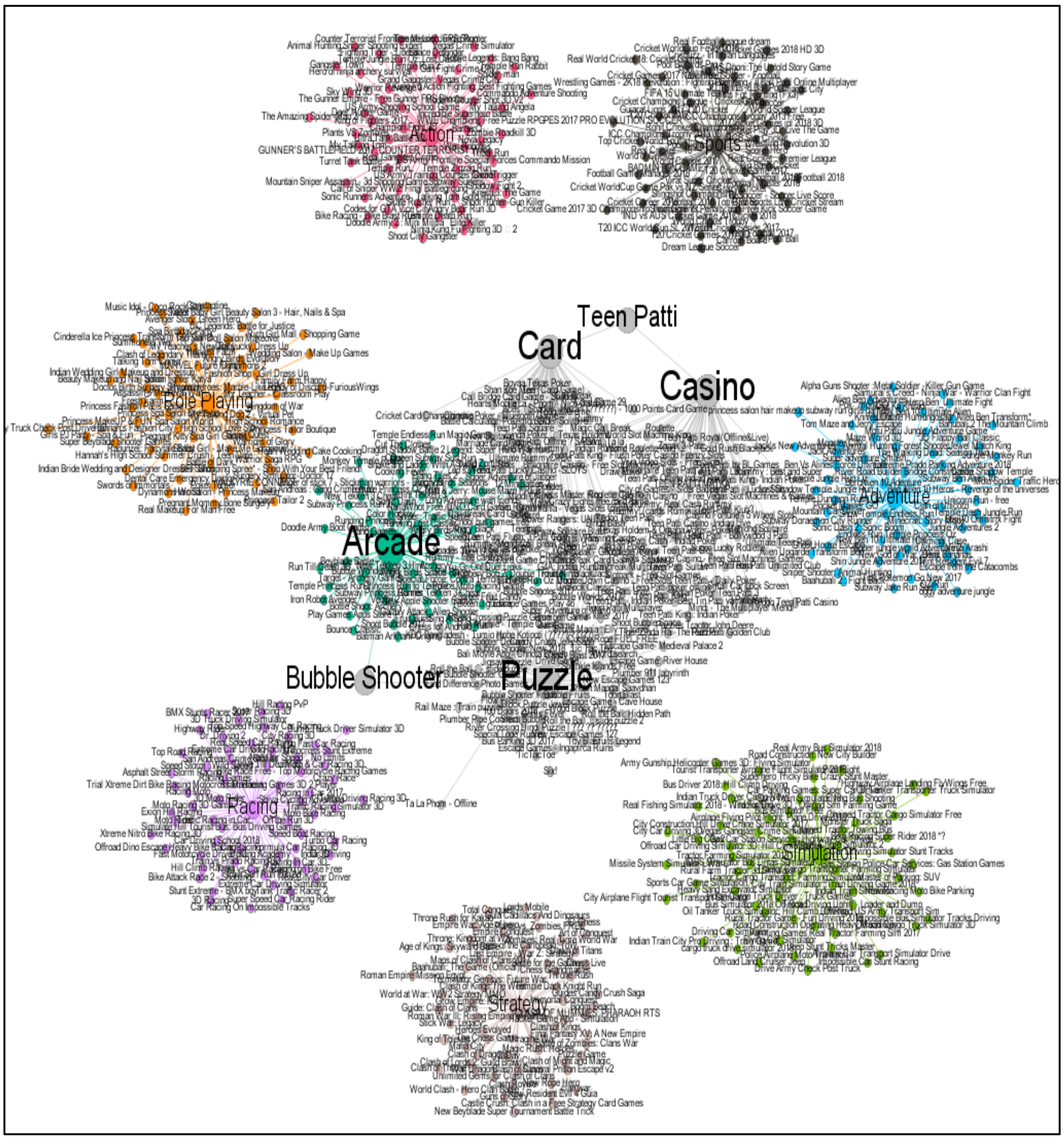


Figure 2.5 Interaction Network of Mobile Games based on category prepared from 9Apps.

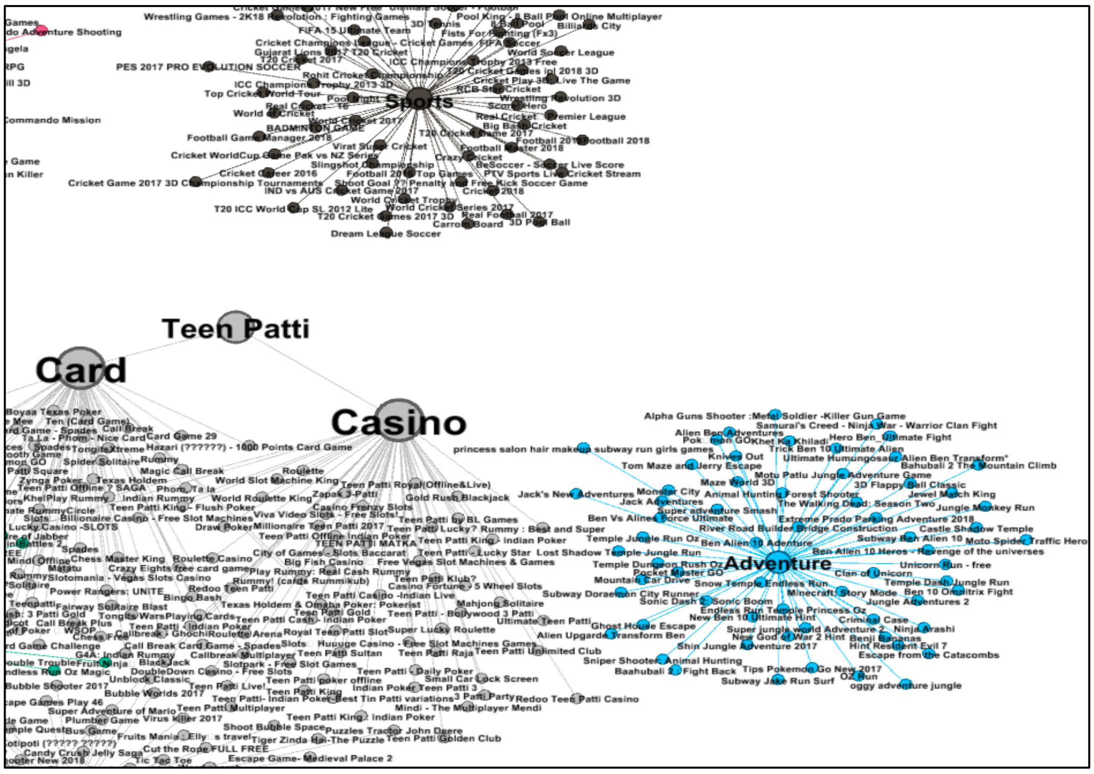


Figure 2.6 Focus on *Teen Patti* in Interaction Network of Games based on category prepared from 9Apps

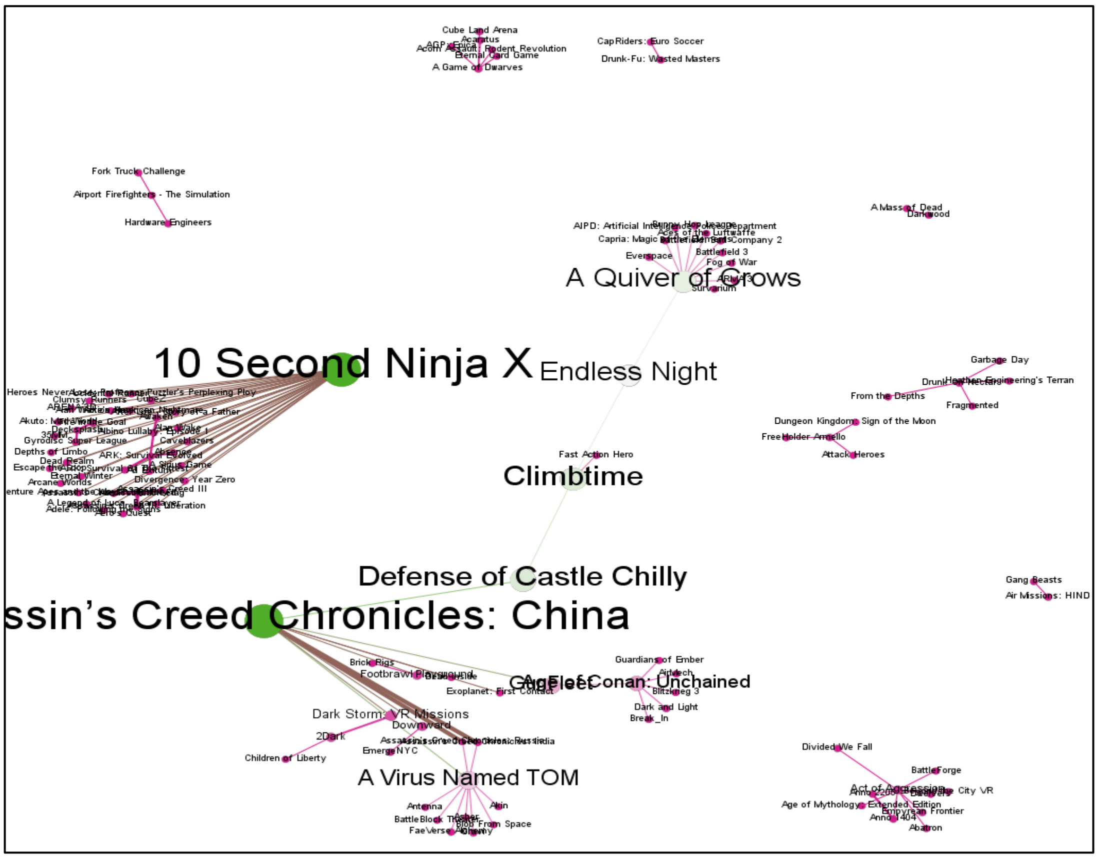


Figure 2.7 Interaction Network of Computer Games

Kneifer and John [13] studied the psychological impact of violent games on the tender brains of kids. The study has been carried out using SNA techniques like degree distribution, centrality measurements and community detection on the interaction network of games.

Realizing non-social domains as social networks opens doors to several possibilities of having more insightful applications of Social Network Analysis (SNA). Building interaction network of entities from non-social domains is transforming the entire domain into a social network having exactly same structural characteristics as possessed by any traditional social network.

In current scenario social networks have gained tremendous popularity among every part of common people's lives. This popularity has brought researchers to focus on analytical impact of social networks. Social Network Analysis (SNA) has emerged as a prominent field of study in modern sociologies. Social network analysis is usually carried out on the network of actors involved in any social activity or interaction where actors are defines as active individual members of the network having connections amongst them and primarily executes in two phases, data collection [14, 15] and data analysis [16, 17]. Next section discriminates data collection from normal web and data collection from social networks.

2.3 SOCIAL WEB CRAWLING

A crawler is a program which basically traverses the network to collect data from various nodes or sites. It starts from a seed and proceeds further with the help of links present at the seed. The seed is the starting web page from where the crawler beginss crawling process. Traditional web crawlers are generally used to collect data for indexing purposes which is generally used by search engines, whereas social web crawlers focus on collecting the structural properties of the social network. Social web crawler starts with a particular user which is seed node for the crawler. Then it explores its friend list and finds more nodes to be crawled further. The traditional crawler for normal web starts with a seed URL and collects all the information present at this seed URL page and extracts URLs from this content. These new URLs found at the current crawled page are added to to-be-crawled-next list (URL frontier) of the crawler.

Both the above crawling strategies are same in terms of basic concept but different at implementation because normal web page and a page of social network have different structures and different content to be crawled. In case of WWW, all the content present at the page is downloaded and the content is used for indexing the web pages. In social web page the content lies in different categories, e.g. the friend list of the user containing new users for the crawler to be crawled further, general information of the user like hometown, study place, work place etc., the content posted by the user, the pages/groups/communities liked or joined by the user etc. The crawler may be interested in one or many of such categories.

Figure 2.8 And figure 2.9 Depicts the working architectures of a traditional web crawling and social web crawling.

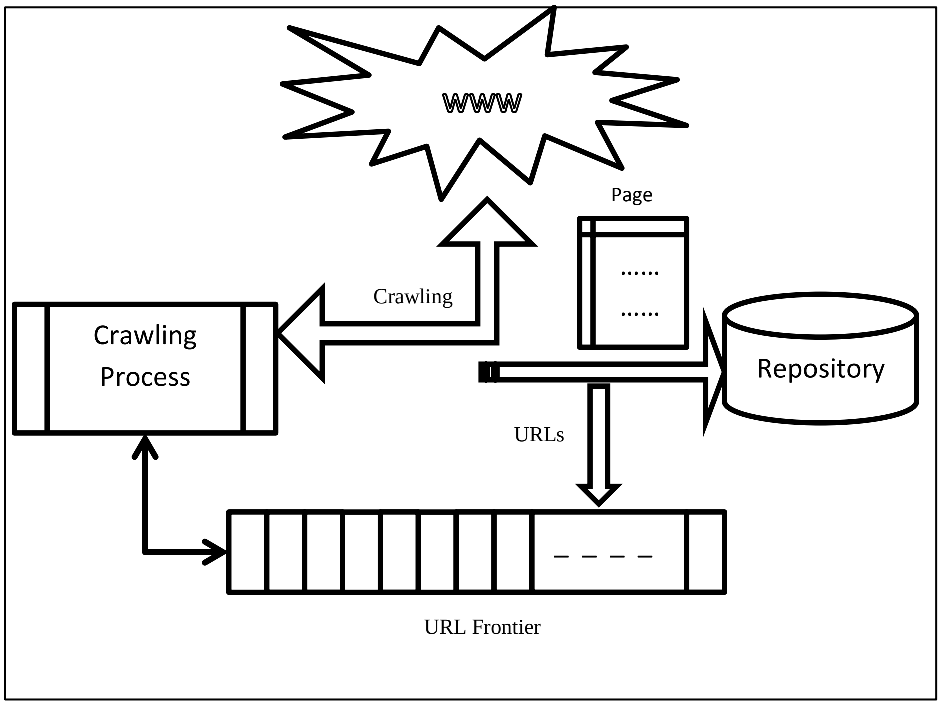


Figure 2.8 Traditional Web Crawling Architecture

Another difference between a conventional web crawler and a social web crawler is the protection/security checks the crawler has to deal with. Normal web crawlers do not face much resistance because there are not many options available to block a crawler on a web site. One way to block a crawler on a website is protecting the website by password. Deep webs, where the content is hidden behind a form, are also hard to crawl. Another way is blocking IP addresses of every crawler provided the IP addresses of every crawler is known (which may or may not be a big deal). On the other hand social web crawlers face bigger challenges such as, almost every social web is password protected. Crawler requires valid user name and password to enter into social web. Moreover, the users on social sites get security features like hiding their friend lists, making their posts private; hiding the pages/groups/communities they have joined, hiding the personal information, protecting their profile pictures etc. A well protected user is dead end for social web crawler.

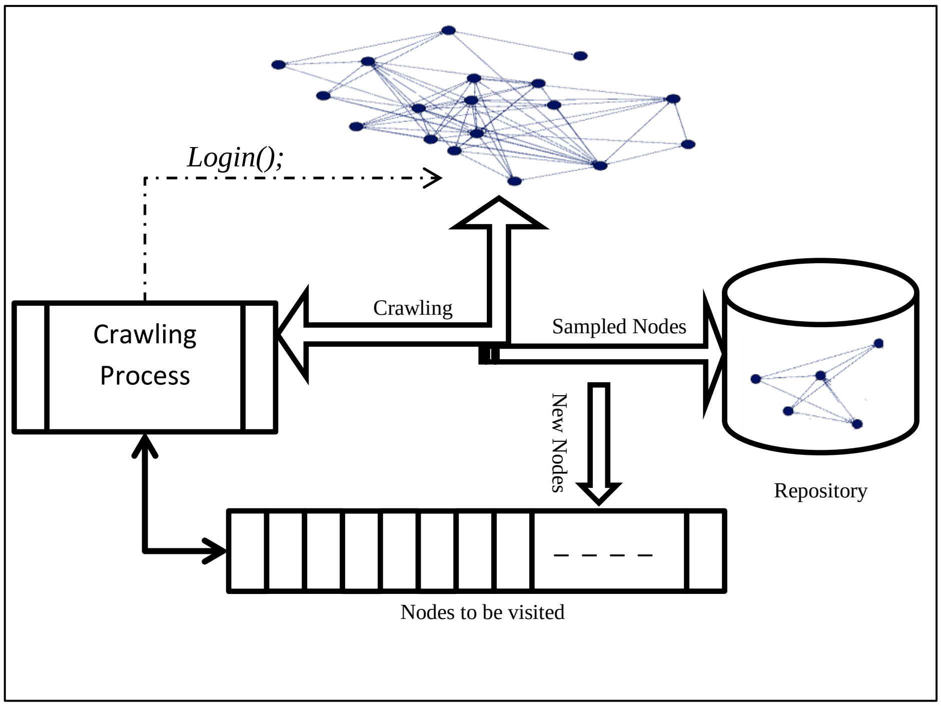


Figure 2.9 Social Web Crawling Architecture

Although, OSNs provide easily available data for analysis, the size of OSNs is gigantic that hinders researchers to understand the structure of graphs. Huge size of OSNs brings the challenge of collecting information from complete graph. The primary reason is the reluctance of administrator to share the data or the users on the site have different restrictions on visibility of their data. Time required to acquire complete graph makes it impossible. Secondly, if somehow the data of complete graph is gathered at one place, it requires expensive and well-equipped computers and large overhead in terms of time, storage and computation [12]. Alternatively, sampling of graph suggests a prominent and inexpensive solution. A subset of graph is considered representative of the original graph. A good sampling cuts short the scale of the original graph, yet maintains its characteristics.

The social network data crawled by social web crawler is stored in the data set in the form of adjacency matrix or adjacency list. Along with the users and their associations, other information of the actors in the form of their attributes can be collected by social web crawler.

Social web crawlers face two major challenges, first is the gigantic size of the social network. To deal with gigantic size of social network, crawlers prefer to collect a sample of social network instead of crawling complete network. Sampling has an added challenge in it which is called 'biasing'. The crawlers may get biased towards similar kind of nodes. If the sample contains all similar nodes then then it is not a good representative of the social network. Second challenge is dynamic behaviour of the social networks. The sample collected and stored on the local storage may soon get obsolete as the social network is continuously changing. Social web data collection has to handle dynamic behavior of social networks too. Researchers have proposed several significant sampling logics for crawling social networks which have been discussed in chapter 2. All social network analysis procedures are applied on the sample of the actual social network. The conclusions are considered true for the entire social network. The next section lists various applications of SNA elaborating the importance of SNA. Subsequent sections throw light on the techniques and metrics of SNA.

2.4 APPLICATIONS OF SNA

Social network analysis has wide range of applications in various areas. Some of the recent areas of applications have been pointed out by Krebs [19] as follows:

- Examining spread of disease among cows in a farm by analyzing the network.
- Faculties of various universities create diverse range of communities on the basis of their interests.
- By preparing network of researchers from research publications, information flow can be revealed.

Applications of SNA in various fields are possible due to availability of ample amount of social data on online social network. But there are certain challenges and issues related to the crawler for social networks, which are highlighted in work mentioned in [15]. It has emphasized upon the aspects of biasing, efficiency and sensitivity of crawler. Biasing results in a skewed sample collected form social network.

A very unconventional and prominent field of SNA application is combatting terrorism. SNA techniques have been successfully identifying and controlling the terrorist activities and outfits. Terrorist organization have a specific structure for spread of radical ideas and recruitment of people [46]. SNA techniques help in identifying these organizations and critical information can be obtain to detect and predict terrorist activities. A novel approach has been proposed by Hussain [47] that says if main lead of the is identified and removed then it reduces the strength of their network.

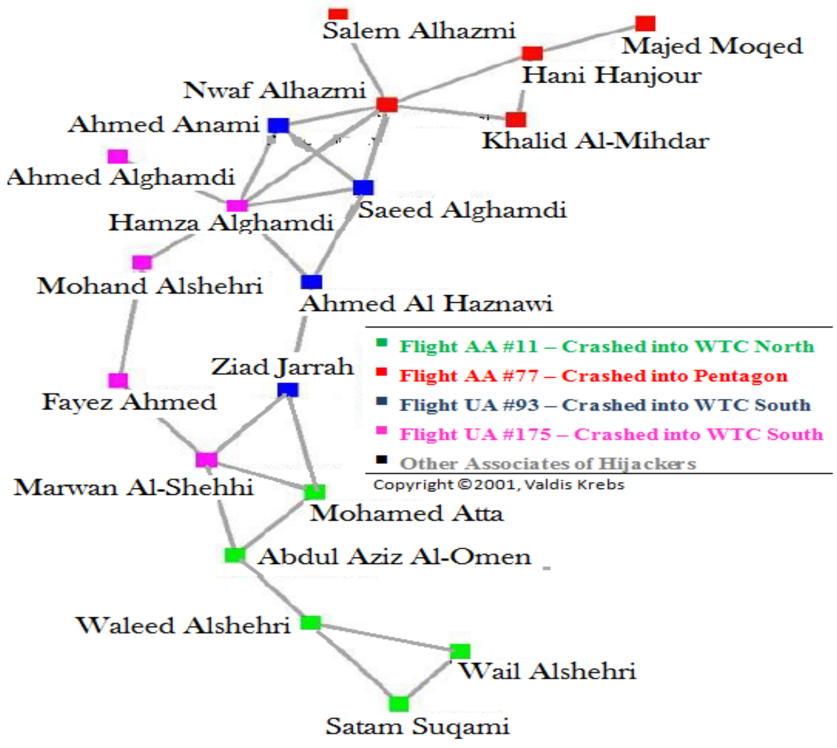


Figure 2.10 The network of the 19 hijackers involved in 9/11 attack.

It is a two stage framework. In first stage every node of the network is assigned an uncertainty weight and in next stage an argument driven hypothesis is applied to rectify the assumptions. Carley [48, 49] used simulation and predictive analysis based complex modelling tools for the research and analysis of terrorism. DyNet is a tool which transforms the text data into a pictorial network in which the network can be destabilized gradually. Other tools like Automap [48] and NETTEST [50] are used to represent the terrorist networks and groups. Terrorist organizations are using social networks as one of the most prominent platform to spread hatred. These tools help in analysing the social media. Figure 2.10 shown the network of terrorists involved in 9/11 attack prepared using SNA techniques.

Weinstein et al. [51] has proposed a counter terror social network technique based on intent recognition method. It uses identification and exploration of tools used in dynamic social networks to detect and track terrorist activities in the network.

Another prominent area of application of SNA techniques is organizational management. By mapping the collaborative work of the researchers and workers, individuals with similar ideas can be identified. A keyword based mapping is done to identify the people of same interest [52].

A network of doctors and practitioner is created and analyzed frequently to identify the prominent disease for taking significant precautionary actions in advance. This network is also used for medical referral process [53].

Clustering the web is another prompt application os SNA technique [54]. Partitioning of the web is done using the search log. Search log contains, search query and the resultant URLs. The similarity between web pages is identified and the network is created. Then clique partitioning method is used to identify the clusters of the web. The connection between websites depends on the similarity threshold set in advance. Web clustering is used for supervision of WWW as it can detect spam, pornographic websites, websites spreading radical ideas or political hatred etc.

Page-Rank algorithm [55] and HITS [56] algorithm for assigning significance weights to web pages are also applications of SNA. The WWW can be seen as the social network of web pages where the pages act as actors and hyperlinks present on the web pages are the basis of their association. That means a web page will have a directed edge connected to every other web page whose hyperlink is present on this page. Page Rank algorithm and HITS both make use of this network of web pages.

Blogs have gained popularity in recent past. Blog is a website where the user (blogger) posts articles which are maintained in reverse chronological order. These blog entries have links to other web pages which are especially blogs web sites. This whole network is called blogspace. Its structure is almost similar to any social network and therefore, some researchers have integrated SNA with blogosphere [12]. As subset of WWW, weblogs are also seen as social networks, despite the fact that they are different from blogs in terms of entry to other hyperlinks and comments.

Researchers have put extraordinary efforts to make web understandable, meaningful, fit for machine processing [57, 58]. Such web is called semantic web. The objective is bring everything around knowledge centric outlook. Social networks and semantic web are complementary to each other and support each other. Trust networks [59, 60] are semantic web enabled social networks which a new paradigm is created for users to outsource beliefs and knowledge through social networks.

Information management [61] and disaster management [62] are other prominent areas where SNA techniques are used with utmost attainment.

The above discussed few significant applications of SNA establish the fact that SNA techniques are gaining popularity in every field. The reason is advancements in computers for accomplishing analytical and data processing tasks requiring high computations power and advancements in internet due to which online social networks and online social activities of individuals, real time social network data can be obtained easily. Efficient SNA tools and softwares for social network data collection and analysis have been developed by researchers to make SNA more efficient and automated process. Following sections discuss the work done by the researchers in various sub fields of SNA like social network data collection, maintaining freshness, interaction networks, community detection and cliques.

2.5 ASPECTS OF SOCIAL NETWORK ANALYSIS (SNA)

2.5.1 Degree

Degree of a node is defined as the number of nodes adjacent to this node. Nodes which are directly connected are called adjacent nodes. If the graph is undirected the number of incident edges is the degree of nodes. If the graph is directed then the node has in-degree and out-degree as number of incoming edges on this node and number of outgoing edges from this node respectively. In social networks degree distribution has a vital role to play on the structural properties of the network. The degree distribution in social networks follows power law as shown in equation 2.1.

$$P_k \propto K^{-\alpha} \tag{2.1}$$

Where, P_k is the probability that a node has a degree *k*, and α is a constant. α ranges from 1.6 to 3.0 [22].

2.5.2 Geodesic Distance and Diameter [23]

Geodesic distance between two nodes is defined as the shortest possible path between those vertices. This definition holds for both directed as well as undirected graphs. The diameter of the network is the longest geodesic distance between any two vertices in the network, provided the network is fully connected. The geodesic distance and diameter are significant parameters in analytical procedures to understand structural characteristics of the social network. There is a possibility that two nodes may be connected through more than one path between them but if the objective is to know how the relationship between two actors is going to avail opportunities of constraints, then all of the connecting possibilities are not significant. The diameter of a network tells us how "big" it is, in one sense (that is, how many steps are necessary to get from one side of it to the other). The diameter is also useful quantity in that it can be used to set an upper bound on the lengths of connections that we study.

2.5.3 Cliques and Subgroups [23]

Partitioning the actors of social network into cliques or subgroups is an important aspect of social network analysis. It helps in understanding structural behavior of the network. A clique is defined as complete subgraph of the network. In other words, clique is a group of nodes in which every node is adjacent to every other node. A graph may contain several cliques. The size of clique is denoted by the number of nodes in the clique. That means a clique having k vertices is called *k-clique*. The clique of a graph having maximum number of nodes is called *maximum clique*. Finding maximum clique in a graph is a well-known problem of graphs and it is computationally NP-Hard [24].

The definition of clique is very strict which may be too hard for several applications. The definition of cliques is relaxed in a variant of clique called *N-Clique*. In N-clique the nodes are not required to be connected directly but every distance between any two nodes in N-clique should not be greater than N. But there is a possibility that the nodes falling in the path of two nodes may not be part of the N-clique even if the distance between these nodes in N. To eliminate this possibility, another restriction is introduced which says the the intermediated nodes should also be part of the clique. Such cliques are called *N-clans*.

Another variant of clique is *K-Plex*. It says that any node is part of the clique (k-plex) of size n if it has direct ties with *n-k* nodes of the clique.

Another approach is *K-Core*. It says that a node is part of clique of any size if it is connected to k nodes of the clique. K-core approach is more inclusive as compared to k-plex. If the size of k decreases the sub-group size will increase.

2.5.4 Maximum Flow

Maximum flow is a notion to represent the strength of ties between two nodes. Suppose a node X wants to send a message to another node Y and there is on one node B to whom X can send the message to reach to Y then it's a weak connection between X and Y even if B has several ways to reach Y. In contrast to this if X has four such nodes to which it may send the message to reach to Y then this connection is stronger than the previous one. The 'Flow' denotes strength of weakness of ties which directly depends on the alternatives.

2.5.5 Centrality

Social network analysis uses centrality measurement as one of the most significant SNA techniques. Centrality is the measurement of significance of a node in the network. A node is considered significant if it is in center. For instance, in figure 2.11, node X is considered to be in center. Which implies that X has more power to influence people around itself [25].

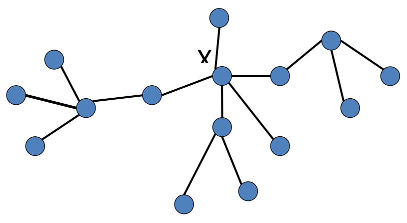


Figure 2.11 Central Node in a Network

The criterion for a node to be in central has several different perceptions. Most widely accepted and used approaches are degree centrality, closeness centrality, and betweenness centrality.

(A) Degree Centrality

This the simples approach of centrality measurement. The node which has largest number of nodes directly connected to it is considered to be most central. In other words node having highest degree is most central. The centrality of v in a network having *n* nodes is defined in equation 2.2.

$$C_d(v) = d(v) \tag{2.2}$$

Where, *d(v)* denotes degree of node *v* and $\boldsymbol{C_d}$ denotes degree centrality of node *v*. In a simple graph, degree centrality of a node ranges from *0* to *n-1*.

Degree centrality represents true centrality of a node but the only problem in equation 2 is that the degree centrality of nodes in different graphs cannot be compared. To have a comparison of degree centralities of nodes on different scales, normalization is required. Equation 2.3 represents normalized degree centrality of a node.

$$C_D(v) = d(v)/(n-1) \tag{2.3}$$

(B) Closeness Centrality

The limitation of degree centrality is that it only considers direct ties of the node to measure centrality. Closeness centrality considers direct as well as every indirect tie of a node to rest of the nodes. Closeness centrality takes into account the sum of geodesic distances of every node from this node. For a network having n vertices and closeness centrality of node $\boldsymbol{v_i}$ is measured using equation 2.4.

$$C_c(v_i) = \left[\sum_{j-1}^{n} d\left(v_i - v_j\right)\right]^{-1} \tag{2.4}$$

Where, $\boldsymbol{d\left(v_i - v_j\right)}$ is the geodesic distance of $\boldsymbol{v_j}$ from $\boldsymbol{v_i}$ and $\boldsymbol{C_c(v_i)}$ represent closeness centrality of node $\boldsymbol{v_i}$. Equation 2.5 denotes normalized closeness centrality of node $\boldsymbol{v_i}$.

$$C_C(v_i) = (n-1)C_c(v_i) \tag{2.5}$$

(C) Betweenness Centrality

If two nodes are not directly connected then definitely they are going to affect each other through some other intermediate nodes lying on the path between these nodes. The idea of betweenness centrality says that the most influential node is the node which is intermediate node for maximum number of pair of vertices. In a network having n vertices, the betweenness centrality of a node v is measured using equation 2.6.

$$C_b(v) = \sum_{j<k} g_{jk}(v)/g_{jk} \tag{2.6}$$

Where, $\boldsymbol{g_{jk}}$ is total number of shortest paths in the network and $\boldsymbol{g_{jk}(\mathbf{v})}$ is the number of shortest paths where v is one of the intermediate nodes. Equation 2.7 represents normalized between ness centrality.

$$C_B(v) = C_b(v)/[(n-1)(n-2)/2] \tag{2.7}$$

In figure 2.12 node X has highest betweenness centrality whereas, Y has the highest degree centrality.

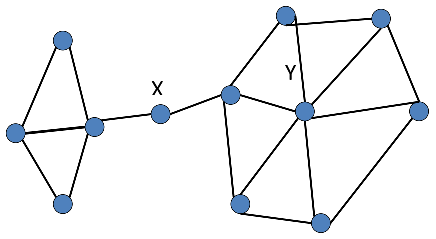


Figure 2.12 Betweenness Centrality v/s Degree Centrality

2.5.6 Power

Power is a fundamental characteristic of social structures. Every researcher agrees to this fact despite there is a dispute on what power is. Basically it is closely relate to the centrality measurements. Table 2.1 provides a summary of study of power with respect to centrality.

Table 2.1 Comparisons of Power Aspects

Power Aspect	Description	Impact
Degree	Number of direct ties	More alternatives for flow.
Closeness	Distance from other nodes	Opportunities to bargain.
Betweenness	Being intermediate node for pair of vertices.	Having power to disconnect nodes.

2.5.7 Small World Effect [22, 26]

Small world effect is an observation of influence of social networks. The observation says that the distance between two actors is much smaller than the size of the network.

Social networks especially online social networks have brought people closer and have reduced the size of the world.

2.5.8 Clustering Coefficient [22, 26]

Clustering coefficient is a significant measurement in social networks which is widely used in community detection techniques. A triplet in a graph is defined as group of three vertices having atleast two edges. For instance, consider a triplet $< i, v, j >$, having atleast edge $< i, v >$ and edge $< v, j >$. This triplet is open triplet if it has on two edges and it is closed triplet if it has all three edges ($< j, i >$ also). Clustering coefficient of a vertex v is calculated using equation 2.8.

$$C_v = \frac{number\ of\ closed\ triplets\ centered\ around\ v}{number\ of\ triplets\ centered\ around\ v} = \frac{T_v}{d_v(d_v-1)} \qquad (2.8)$$

Where T_v is the number of closed triplets around v and d_v is the degree of v.

SNA is identifying above mentioned characteristics in a graph and draw conclusions based on the objectives of the application. Along with these parameters, most widely used SNA technique is community detection. Next section discusses community detection in social networks.

2.6 COMMUNITY DETECTION

Community is a cohesive group of nodes that are more closely connected to each other than they are with rest of the nodes. Community detection is identifying such communities in the social network. Community detection is an important tool for social network analysis. Due to increase in usage of media and internet, large social networks emerged which have complex structures containing communities and clusters. These communities are part of the network and influence the structural characteristics of the network. Therefore, detecting communities is a powerful mechanism to understand the behavior of the actors of the network [27]. The researchers have developed several methods to study overlapping attributes of the actors and to find communities in a social network. Community and clique have almost similar definition.

A clique is a complete subgraph of the network. Finding a clique having maximum number of nodes in a graph is called Maximum Clique Problem [12]. Due to similarity in clique and community, community detection is one of the most significant applications of cliques [28, 29, 30]. Community detection in a social graph is computationally intractable problem because the social graphs are gigantic in size. This is one of the challenges faced by researchers working on community detection techniques. Another challenge is dynamic behavior of the social networks. The communities detected at one point of time in a social network may not be present in the network after some time.

2.7 SUMMARY

In this chapter, aspects related to Social Network Analysis (SNA) have been discussed at their terminological level. SNA has gained popularity due to increase in the usage of internet. In present scenario almost every person is directly or indirectly part of one or more social media platforms. The people on social networks are connected with each other and share their interests and opinions freely. The major objective of SNA is to study the structural properties of the social networks and derive conclusions on the behavior of the actors of social networks Online social networks are generally gigantic in size and dynamic in nature. Social network data collection is done using social web crawlers. Social web crawlers face huge size of the network as major challenge. Instead of collecting complete network a sample of the original social network is preferred to be collected by the social web crawler. Social networks are not limited to have people in it but non-living things can also form social network. Non-social domains can also be realized social networks based on the interaction between their entities. Such social networks are called interaction networks. An example of interaction network prepared from domain of computer and mobile games has been discussed. Social network analysis includes tools, techniques, softwares and procedures to determine structural properties of the social network such as degree distribution, centrality measurements, modularity etc. The components on which SNA focuses during analysis of social networks have been discussed in details along with their impact. Most commonly used social network analysis technique is community detection. Communities are defined as denser area in the network, which means the nodes within a community are more closely connected than they are to rest of the nodes. Clique is also similar, as in clique every node is connected to every other node of the clique (complete subgraph). Therefore, cliques have a significant application in community detection. Community detection faces two major challenges; first is gigantic size of the social graphs and second is the dynamic behavior of the social networks. To handle huge size of the social networks, various optimizations are applied by the researchers to male community detection faster. To handle dynamism, the social network is considered in snapshots. A snapshot is static image of the social network at a particular time. Lastly, various applications of social network analysis have been outlined such as terrorism detection and prevention, organizational management, epidemic disease, medical referral system etc.

In the next chapter, significant works of researchers and engineers on various aspects of social network analysis have been discussed. The objective of the discussion of literature in this area is to identify significant contributions of the researchers and to identify the major challenges in the field of SNA.

CHAPTER 3

RELATED WORK

3.1 INTRODUCTION

Social network analysis has significant results based behavioral analytics applied on social networks. The reason behind this is huge interest of people towards social networks. People are tend to interact and therefore prone to be on social networks. The ideas, interests, orientations etc. shared by people on the social network creates a strong base for behavioral analysis of the targeted people. Due to this real time data availability and power to impact lives of people directly, Social Network Analyzer (SNA) has applications in numerous fields as discussed in chapter 2.

The main objective of SNA is to collect real time data available on social networks and to identify the structural properties of the social graph. Gigantic size and dynamic behavior of the social network makes data collection as well as analysis, a great challenge.

The researchers have addressed the challenges related to SNA in their contributions. The further discussion in this chapter is focused on significant contributions of various researchers. The objective of further discussion is to understand existing solutions of the challenges, their achievements and limitations.

3.2 SOCIAL NETWORK DATA COLLECTION

Although, the social network data is easily available on online social networks but the gigantic size of OSNs is a fair challenge for researchers. Huge amount of data present of OSNs requires almost infeasible time to collect it. If somehow the time is not a constraint then the computing and storage capacity required for such huge amount of data is a bigger challenge to handle [12]. As an alternate, sampling of the social networks is done, in which instead of collecting complete social network, a representative sample of original social network is collected which is much smaller than the original social network. Earlier efforts made by researchers for sampling large social networks are BFS and snowball techniques [63]. The social web crawlers use these as sampling logic. These techniques start at a seed node and then autonomously migrate from node to node through association links and capture the sample in the local dataset. These techniques are major victim of the well-known anomaly of sampling called '*Biasing*'. The sampling algorithm tends to crawl similar type of nodes, mostly nodes with higher degree. The sample collected by the technique which is biased towards one kind of nodes shall not have structural and statistical characteristics of original network [65]. A major portion of work done to handle biasing and to develop unbiased sampling technique is based on random walks in the social network [67, 68, 69, 70, 71, 72].

Multiple algorithms proposed by various researchers in the literature can be put in two categories: techniques that keep nodes in the centre and the techniques that take edges into account. BFS [63, 64], MHRW [63, 70, 71] and USDSG [75, 76] are significant algorithms which fall into first category whereas Frontier Sampling (FS) [75, 77] belongs to second category.

BFS is most widely used sampling technique for understanding behavioural orientation of users of OSNs [63] and to analyze social networks for their topological characteristics [66]. But it is majorly affected by biasing [63, 66]. BFS gets attracted towards higher degree nodes. Therefore, the sample collected by BFS has higher local clustering coefficient as compared to the original network.

Metropolis Hasting Random Walk (MHRW) selects nodes randomly on the basis of their degree distribution probability [70, 71]. MHRW is based on the Markov Chain Monte Carlo (MCMC) model of probability distribution. A proposal function is designed in MHRW on the basis of probability distribution to select and reject the nodes randomly. The probabilities are continuously modified during sampling to achieve probability distribution convergence. Another event called random number generation directly affects selection of the nodes. A random number p in the uniform distribution of 0 to 1 is generated. If the ratio of probabilities of current node and the node which to be selected is greater than p then the proposed node is selected. In this way a node with lower degree has higher chances of getting selected. This is reverse of biasing. MHRW was initially designed and used for undirected graphs. Unbiased Sampling in Directed Social Graph (USDSG) [76] is an extension of MHRW for directed graphs. USDSG can also be used for undirected graphs by considering the edges to be bidirectional. USDSG is applicable on the graphs which symmetric.

Therefore, the graph is first converted in to symmetric graph. The similar methodology is also used in Frontier Sampling (FS) [76]. FS takes decision of selecting the nodes based on edges. It also uses principles of random walk. The probability of selecting a node *u* from node pool *N* is defined in equation 3.1.

$$P(u) = \frac{k_u}{\sum_{v \in N} k_v} \tag{3.1}$$

It selects an outgoing edge of *u* i.e. *(u, v),* uniformly and *u* is replaced by node *v* in pool of nodes *N*. This edge becomes part of the sample. The performance of FS is not good if the clustering coefficient of graph is small [75].

An event driven sampling technique has been proposed by Corlette et al. [73] which takes into account the active part of the social graph. The idea of event driven sampling is same as the technique used by search engines to maintain freshness of their repository (eliminating expired pages and adding fresh pages). Multigraph sampling technique [74] focuses on clustered view of the social network. It considers social network as set of nodes distributed in clusters. Multiple relations of nodes among themselves are used as basis for sampling the nodes.

A social network may have multi-nodes, multi links and associations at multi-level [78]. Multi node means the network may contain multiple types of nodes e.g. people, products, locations etc. Multiple links between a pair of users may be present in the network. The extent of multiple links among nodes may be different

A focused crawler for facebook has been proposed by Leunge et al. [79] which exploits the profile data of user with the aim to enhance security of personal computer and general clients. HTTP packets are monitored to understand the communication between client and facebook server. The identification of fake profiles is carried out by this monitoring. A BFS based social web crawler has been proposed by Catanese et al. [80] in which uniform sampling [67] is used for selection of nodes. The problems of duplicate links, uninformative pages and duplicate content on web has been tackled by graph pruning based on web clustering in [81].

Dynamic behavior of social networks brings another major challenge for collection of social network data as well as for analysis for social network data. Due to dynamism of social graphs, the sample present in the local repository may not always be a good representative of the original network as its quality is subjected to the evolution speed of the social network. Social graph dynamism can be handled at social data collection level as well as at the time of analysis of social network data. Following text specifies few works of literature for data collection using focused crawlers.

Since the web has gained popularity, researchers started exploring different ways to gather information from web pages which is later used for indexing the web pages and aiding the search engines. Focused crawling is one of the most popular approaches to do this. Researchers started grilling on link contexts in web pages. Link context are used for topical crawling, classification of pages and search engines. Anchor text is used by McBryan [82] for indexing the URLs in www worm. Similarly in Google Search Engine anchor text is used to index pages [57]. Craswel et al. [83] proved effectiveness of anchor text for making the web pages for a search engine solely based on pages indexed by link contexts. Topical locality of web is tested by Davison [84]. Web pages contain hyperlinks to the pages with similar context.

Topical crawlers generally rely on the context of hyperlinks. Topical crawling works in [85] and [86] are influential link context works. Iwazume et al. [87] used ontology along with anchor text. To guide topical crawlers, various heuristics are also used. Topical crawling works [88, 89, 90] have used standard information retrieval techniques such as cosine similarity to queries, profiles or on topic examples.

Bedi et al. [91] proposed a multithreaded Semantic Focused Crawler (SFC) which utilizes domain ontology to specify the topic and decide seed URLs. Dong and Hussain [92] also made use of semantic focused crawling and ontological learning by designing an unsupervised framework for vocabulary based ontology learning and designed a hybrid algorithm for matching semantically relevant topics. Yazun Due et al. [93] designed seed URL selection approach based on user interests ontology is used to expand topic queries. Sheng Yuan Yong [94] used ontology supported website model which provides solution to an information agent i.e. ontology supported information shell.

The above discussion has reviewed significant work done to collect data from social networks. The conclusion from this discussion states that the major challenges for social web data collection are gigantic size of the network and biasing of crawler towards higher degree nodes. Researchers have contributed significant efforts to handle these challenges. The dynamism of the graph is hard to handle at social web data collection phase. It is handled at analysis phase (community detection) of social network data which will be discussed in next section. Some significant work done in the field of focused crawling has been discussed above. The next section discusses significant work on community detection in a social network.

3.3 COMMUNITY DETECTION

Community detection in a social graph is to identify the denser sections of the graph where the nodes are more closely connected to each other as compared to their connection with rest of the nodes of the network. Community detection is most widely used SNA technique which has applications in various fields. Work done by researchers for community detection can be categorized into two categories – first, detecting communities as disjoint sets and second is detecting overlapping communities. Community detection techniques falling in first category are Louvain [95], infomap [96], label propagation [97], Newman's leading Eigenvector mechanism [98] and fast greedy approach [99]. These are discussed in detail later in this section. The qualitative measures for community detection are modularity [100] and conductance [101. In case of disjoint community detection method the nodes are treated individually and assigned to communities one by one on the basis of their connections to other neighboring nodes. In this way a particular node becomes the part of only one community whereas in case of overlapping community detection, a node can become part of multiple communities at the same time depending upon its attachments to other nodes [102].

Overlapping community detection finds more practical acceptance by researchers because in reality a person can be part of several behavioral groups like family, work, well-wishers etc. Wen et al. [103] has proposed an overlapping community detection technique based on maximal cliques. It uses evolution of communities to refine the boundaries of the communities.

There are three components in local community detection paradigm. First is to model the real world graph into a weighted or binary graph. Second is to identify the measures to be used for characterizing the efficiency of the communities. And third is to develop an algorithm for community detection which gives results up to the decided efficiency metrics. Cui et al [104] has categorized these methods as community detection technique with maximal constraints. The other category is of community detection techniques with predefined threshold constraints. Wu et al. [105] has suggested performance metrics to be used in community detection based on the approach used for defining communities in the graph. For instance, if densely connected nodes are assembled for community detection then the performance metrics used to check efficiency of algorithm and quality of communities are classic density [106], edge surplus [107], and minimum degree [108].

External sparseness and internal density attributes are optimized in second category. In this category, modularity of subgraphs [109], density isolation [110] and conductance [111] are used as measure metrics. The third category focuses on the nodes at the boundary of the communities. These nodes are assumed to be connected more closely with the nodes within the community as compared to the nodes outside the community. The sharpness of the boundary is measured and optimized by local modularity [112]. The measures considered above are best for community detection quality assurance but sometimes these measures might impose irrelevant subgraphs on communities containing query node. This is referred as rider effect. Wu et al. [105] eliminated rider's effect in their query biased goodness metric method. Most of the local community detection methods focus on weighted graphs but goodness metric can be modified to work for binary graphs as well. Equation 3.2 is used to calculate density of a subgraph c = (V, E, W), which denotes connectedness of the nodes in subgraph. It is defined as ratio of total edge weights to total number of nodes in the subgraph.

$$D = \frac{\sum_{u,v \in V} w(u,v)}{|V|} \tag{3.2}$$

Where *w(u, v)* is the weight of edge *(u, v)* and |V| is number of nodes in C.

Dynamic behavior of social networks puts another challenge on community detection as the communities detected at a point of time may not hold later. To handle this situation researchers have put efforts in finding the evolution pattern of the communities in social network. The views of social network at different intervals are captured which are called snapshots of the network and evolution of communities is identified. In upcoming section, significant contributions of researchers in this area are discussed.

3.3.1 Community Detection and Evolution in Dynamic Social Network

Significant work to detect communities in social networks has been done in literature such as modularity methods [98, 113, 114], spectral clustering methods [115], stochastic methods [116, 117, 118], and heterogeneous clustering methods [119, 120]. But the common problem that goes unaddressed in these solutions is dynamic behavior of social network. The above mentioned community detection techniques detect a static community across many snapshots. But in reality the social networks change with time and require a technique which considers the dynamism of social graphs.

In recent years, the researchers have started working on evolution of social graphs with time. White et al. [112] proposed a technique for community detection which observes the network with respect to time and over different relations. A study of evolution patterns in social graphs has been done by Leskovec et al. [122] which is based on characteristics of large social networks such as small world effect and degree distribution. The observed pattern is further used to generate the graph which satisfies the discovered patterns. Backstrom et al. [123] proposed a prediction model, which can predict the probability of a node to join explicitly defined communities. The model is based on critical factors which can capture evolution of communities. Kumar et al. [124] analysed the structural evolution of two real world networks and produced certain properties. The identified properties are considered only at graph level and not at community level.

Falkowski et al. [125], first defined the proximity of graph nodes by using graph mining algorithms DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [126] and its variant Incremental-DBSCAN [127]. DBSCAN focuses on denser areas of the network to capture clusters. A node is considered a core point if its neighbourhood has a predefined minimum radius and minimum number of nodes around it. Any other node which is not fulfilling properties of core point or it is not in neighbourhood of any core point, is considered as noise point. Such neighbourhoods are clusters. Incremental-DBSCAN considers insertion and deletion of nodes and edges. DBSCAN and Incremental DBSCAN are generally used for spatial data. This density based idea for graph mining is used in social networks by Falkowski et al. in [125]. The algorithm is called DENGRAPH. DENGRAPH targets to identify communities by identifying clusters in social networks. The clusters are completely based on the proximity of nodes. Let us consider an interaction graph (social graph) G(V, E). Intensity of interaction between two vertices u and v can be defined as shown in equation 3.3.

$$\mathrm{Intensity}(u, v) = \min\left(I_{u,v}, I_{v,u}\right) \tag{3.3}$$

The proximity of u and v is defined as shown in equation 3.4.

$$\mathrm{prox}(u, v) = \begin{cases} 1 & u = v \\ 1 - 1/\mathrm{Intensity}(u, v) & \exists (u, v) \in E \\ \text{undefined} & \exists (u, v) \notin E \end{cases} \tag{3.4}$$

DENGRAPH puts density connected nodes in a cluster encountered during traversal. If the node does not fulfil the criteria of minimum radius and minimum number of nodes around it are considered for next cluster. Rest of the nodes are dropped which do not find place in any cluster.

For dynamic social networks, the interactions (associations) come as a stream. New associations are formed and old associations are dropped. DENGRAPH works in incremental fashion to adapt the dynamic changes in clusters. DENGRAPH exploits the idea of ageing. The edge in the graph continuously loses weight and it is deleted when its weight becomes zero.

Takaffoli et al. [128] has presented an event based community detection and evolution framework. The events are related to the communities as well as to the individual nodes. The communities detected in a snapshot of social network are identified by a flag which is assigned to each node of the community. The flag is basically based on the common attribute among the members of the community. It can be the common idea they all agree upon or political orientation or common workplace etc.

The events in community evolution may be as follows: Suppose a community is detected in a snapshot. A flag is allocated to this community. In the next snapshot if this community persists then flag is still there. Some new nodes can get attracted towards this community and become its members. These nodes also come under the same flag. There is a possibility that another community gets merged in this community in next snapshot. Then the community having more number of nodes will dominate and its flag is assigned to the new community. The community may get partitioned into two or more communities in the next snapshot. Then the partition having largest number of members will take flag of this community and rest partitions are assigned new flags. Finally, a community may get partitioned into such smaller pieces that none of them could be considered as community then the flag is vanished. In every mentioned event, the portion of the community directly affects its identity. The portion is defined as number of nodes *k*. Total 11 events are defined in this work, out of which 4 are concerning to nodes and 7 are concerning to communities as k-form, k-dissolve, k-continue, n-k-split, n-k-merge, k-shrink and k-reform.. The events are parameterized with *k*.

In further work, Takaffoli et al. [129], reduced number of events related to communities, to five only. In this work the focus was only on the evolution of the communities and not on their detection. The communities in a snapshot of the network can be detected using any algorithm. Meta-communities are maintained instead of having flags. Meta-community is a series of communities which are similar. Meta-communities actually observe the evolution of communities. Five events can occur concerning to community evolution viz. form, split, dissolve, survive, and merge.

Meta-community formation and identification of event occurred; depend upon the similarity between various communities at different point of time. The communities detected at different snapshots are said to be similar if they have a certain percentage of nodes common between them. The minimum required similarity between communities is called threshold, denoted by k. k depends on the characteristics of the network. The framework contains module for community detection as well as module for detection of events. The formation, split and dissolve event of the communities is identified as well as the meta-community is formed across multiple snapshots.

A multi-agent based distributed algorithm for community detection has been proposed by Huang et al. [130]. A flock of autonomous agents analyse the network using self-organization and self-aggregation mechanism. An undirected graph G(V, E) is partitioned into k partitions defined as set $P = \{C_1, C_2, \ldots\ldots, C_k\}$, where components $C_1, C_2, \ldots\ldots, C_k$ satisfy $\bigcup_{1\le i\le k} C_i = G$ and $\bigcap_{1\le i\le k} C_i = \emptyset$. P is the community structure if edge density within components is greater than that between components.

The evaluation function F(P) for P is defined in equation 3.5. A is adjacency matrix for graph G.

$$F(P) = \sum_{1\le i,j\le|V|}(1 - A_{ij})g_{ij} \tag{3.5}$$

Where, $g_{ij} = \begin{cases} 1, & v_i \text{ and } v_j \text{ belong to same component of P} \\ 0, & \text{otherwise} \end{cases}$.

The evaluation function calculates F value for each component defined as the number of edges needed to convert a component into clique. As clique is a complete subgraph, the F value for a component will be inversely proportional to the edge density within the component. If the nodes within the component are closely connected then F value will be smaller. Now the problem of deciding the community structure is reduced into an optimization problem in which, each partition has to minimize F value corresponding to each component. The partition with smallest F value will have community structure of the graph. The minimization function is shown in equation 3.6.

$$P^* = \min(\forall F(P_i)) \tag{3.6}$$

P* is the community structure with minimum F value among every possible partition.

The partitioning and calculation of edge density within the partition is done using mobile agents. Agent based system is suitable for clustered network. The limitation of this system is dynamic creation and elimination of agents and the constraint of predefined k.

The above mentioned techniques for community detection consider communities to be standalone. Therefore, the efforts required to detect communities are a bit greater and some of the significant communities may be discarded due to not considering the possibility of communities being overlapping. In upcoming section the potential community detection methods based cliques are discussed. These techniques take overlapping structure of communities into consideration.

3.3.2 Clique Based Community Detection

A) Clique Percolation Method (CPM)

CPM is a k-clique based community detection algorithm which considers overlapping structure of communities as shown in figure 3.1. k-clique is a clique having k nodes. Two k-cliques are said to be adjacent if they share k-1 nodes. The adjacent nodes are identified in incremental fashion to form communities. This is called percolation of k-cliques. CPM is a two-step algorithm defined as finding the maximal clique and them aggregating them on the basis of their adjacency. The maximal cliques can be found using any brute-force approach. The smaller maximal cliques are considered first. The nodes are then added gradually until the clique is maximal and then rest nodes can be discarded.

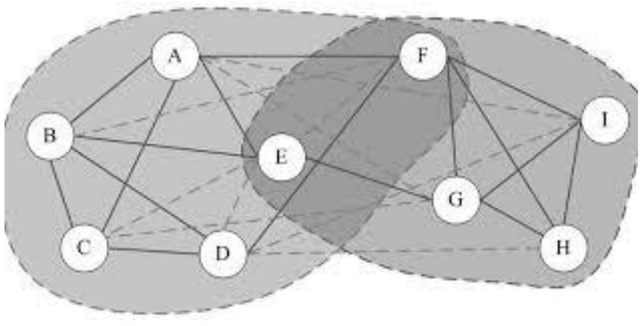


Figure 3.1: A clique percolating with another clique

Derenyi et al. [131] proposed a method which considers random graphs for clique percolation. A threshold value defined in terms of probability denotes the probability of two vertices being connected by an edge. Fundamentally, it is 1/N if clique size is 2. N is total number of vertices in the graph. For generalized k the probability P_c is calculated as shown in equation 3.7.

$$P_c = \frac{1}{[(k-1)N]^{\frac{1}{k-1}}} \tag{3.7}$$

The value of the threshold must be decided carefully because if the threshold is set so high then several nodes would be discarded from being part of the clique. On other hand if the threshold is set too low then undesirable nodes would become part of the clique.

B) Sequential Clique Percolation Method (SPM)

The clique percolation method, Sequential Clique Percolation (SPM) proposed by Kumpula et al. [131] works for weighted as well as for unweighted graphs. The processing time of SPM is scaled linearly because it adds links in the network while examining for the communities. It focuses on the gigantic size of the network and numerous possibilities which are to be thought-about. It explores ingredients of communities. SPM also uses overlapping structure of communities and clique percolation method to detect the communities.

In SPM, the links are added in non-increasing order of load and threshold is verified for the communities. A dendrogram is created to maintain the hierarchy of the communities. The performance depends on the number of nodes and the cliques found in the graph. A dendrogram maintained is shown in figure 3.2. CPM and SPM are tested comparatively on interaction network of proteins [133].

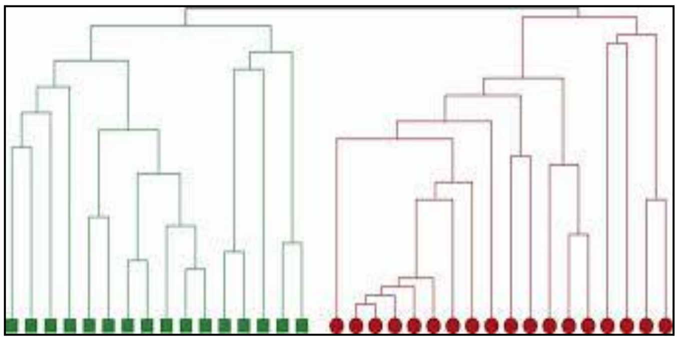


Figure 3.2: Dendrogram

C) Extended Clique Percolation Method (ECL)

Maity and Kumar [134] proposed Extended Clique Percolation (ECL) method as an extension of SPM method. The steps are defined as follows:

- The communities are found using successive steps taking side nodes along.
- The unnoticed nodes are considered to update the communities provided the nodes have common attributes.
- The communities defined over common characteristics are merged together.

ECL uses vertex colouring to identify the nodes having common characteristics. The nodes which do not belong to any cluster are put into a replacement community. Once the communities are furnished and every node is part of any community then the communities sharing common interest are considered as one community. Merging the communities and updating the structure is the final step.

Modularity is used as performance measure for community structure. ECL beats CPM in terms of quality of community structure.

D) Distributed Clique Percolation Method Using MapReduce

Varamesh et al. [135] proposed distributed clique percolation method for community detection in social network, which uses map reduce strategy for distribution. Social networks have gained popularity among every sect of the people. Due to which, even small social networks are very huge. To understand the behavior of the social network community detection is required and to fasten the process distributed clique percolation has been proposed.

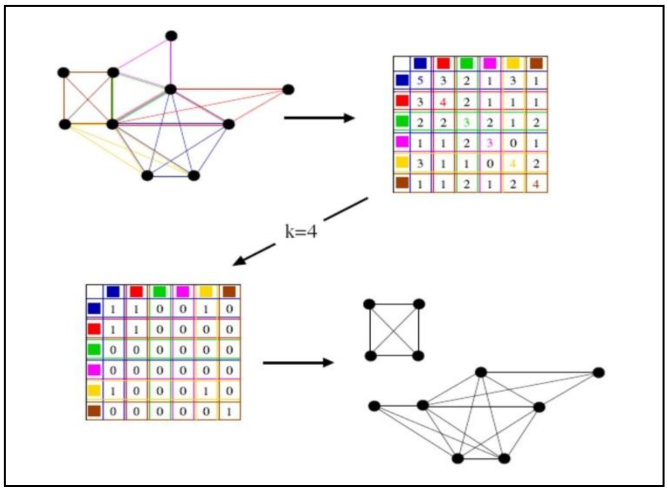


Figure 3.3 The Clique Percolation Method Mapping using Vertex Colouring.

This method uses color coding for clique percolation as shown in figure 3.3. The whole network is divided into smaller parts and distributed among several machines. These parts are processed in parallel. Then at reduce step only those edges are to be considered which have both end points in different part of the network. Figure 3.4 presents general strategy of Map-Reduce method. The clique finding algorithms are run in parallel on every part of the network and once all maximal cliques are found then reduce step takes place. The only overhead of this scheme is IO and memory work load.

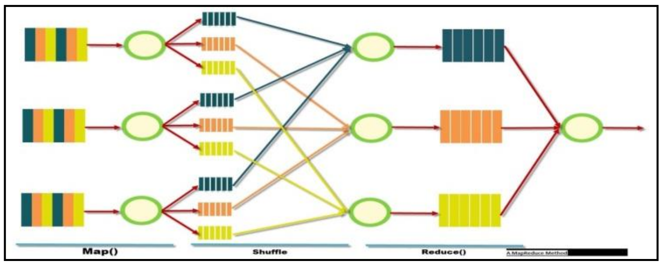


Figure 3.4 Map Reduce Function

Its performance is better than SPM and CPM with a limitation that the number of comparisons in this method is large.

E) Percolation in Complex Network

Reid et al. [136] proposed a community detection algorithm for large and complex networks which is based on k-clique percolation. The challenges addressed in this work are:

- **Large numbers of cliques** are present in the huge social graphs and clique percolation is computationally extensive job.
- **Clique-Clique intersection** operations are required in huge numbers.
- **Individual cliques** are hard to deal with as the cliques are generally considered to be overlapping in large social graphs.

The above mentioned challenges are handled in the proposed algorithm and it outperforms existing k-clique based community detection algorithms like CPM and SPM. The limitation of this method is that it is not able to handle the cliques which have atleast one node in common but do not percolate.

F) Community Detection Using Formal Concept

The efforts of almost every research are focused on reducing the time required to detect communities in the graphs. Hao et al. [137] proposed a k-clique based community detection method for social networks based on Formal Concept Analysis (FCA). Topological impact of the network is enhanced by formal concept analysis. It also helps in understanding the common traits of the communities. On social network the people are strongly connected to each other through friendship of any other behavioural relationship. This method comprises three steps as follows:

- First, the formal concepts from social network are extracted for analysis. A lattice is prepared using adjacency matrix of objects and attributes.
- Second, the relation between objects and attributes is studied through concept lattice.
- Lastly, communities are detected using k-cliques.

This method considers vertices of social network as objects and their relationships as attributes. An Adjacency matrix and concept visualization are maintained to find equiconcepts as shown in table 3.1 and table 3.2. Equiconcepts finding is similar to finding k-cliques. Figure 3.5 represents the concept lattice prepared using adjacency matrix shown in table 3.1.

Table 3.1 Adjacency Matrix

	C1	C2	C3	C4	C5	C6	C7
f1	X			X		X	X
f2		X		X	X		X
f3			X		X	X	X

Table 3.2 Equiconcepts

T	{f1,f2,f3},{c7}
Concept 1	{f1,f2},{c4,c7}
Concept 2	{f1,f3},{c6,c7}
Concept 3	{f2,f3},{c5,c7}
Concept 4	{f1},{c1,c4,c6,c7}
Concept 5	{f2},{c2,c4,c5,c7}
Concept 6	{f3},{c3,c5,c6,c7}
⊥	{ϕ},{c1,c2,c3,c4,c5,c6,c7}

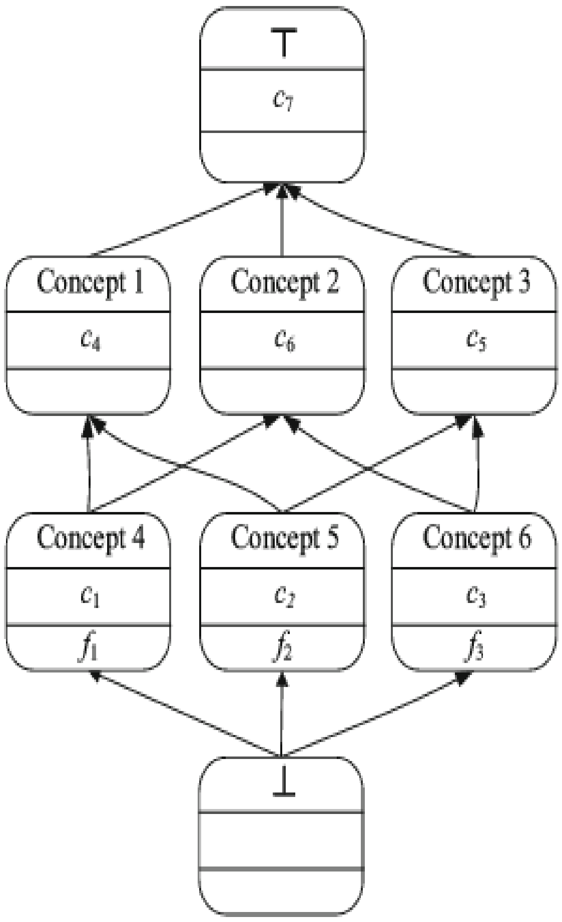


Figure 3.5 Formal Concept Analysis

Interrelation of objects is understood intelligently by FCA. Table 3.3 represents a scenario where different people have their individual interests in breads. A lattice is prepared from this table as shown in figure 3.6.

Table 3.3 Example of FCA

People	**People preferred bread**			
	h1	**h2**	**h3**	**h4**
p1		x		
p2	x		x	x
p3	x		x	x
p4	x		x	
p5	x			
p6	x	x		x

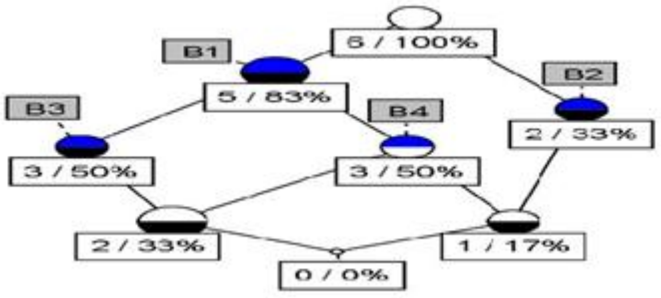


Figure 3.6 FCA Lattice Built Using The Adjacency Matrix.

The algorithm performs better than popular k-clique based algorithms by reducing the computation cost as it is not required to find every k-clique. Instead only the intended concept is targeted. This reduces significant efforts required for clique extraction. The algorithm also reduces computation by reducing overlapping dimensions.

G) Weapon System Architecture Optimization with k-clique community detection

Wang et al. [138] proposed a k-clique based community detection system for weapon system architecture optimization. The objective of weapon architecture optimization is to enhance reusability and flexibility. The existing system faces challenges of interfaces and incompatible views of the workers at the time of attack. To make teams with proper coordination, community detection based system has been proposed. It enhances the relationship among workers and constructive and good knowledgeable workers could be identified.

H) Influence Propagation Model for Community Detection

Alduaiji et al. [139] proposed a community detection method based on temporal interaction among actors which includes network study. The process is depicted in figure 3.7. The motivation for this work is increased interaction among entities not only on the social networks but also in other domains.

There are four steps involved in this method:

- First, the network is partitioned into cliques.
- Second, the edges are assigned weights corresponding to the number of interactions held between nodes.

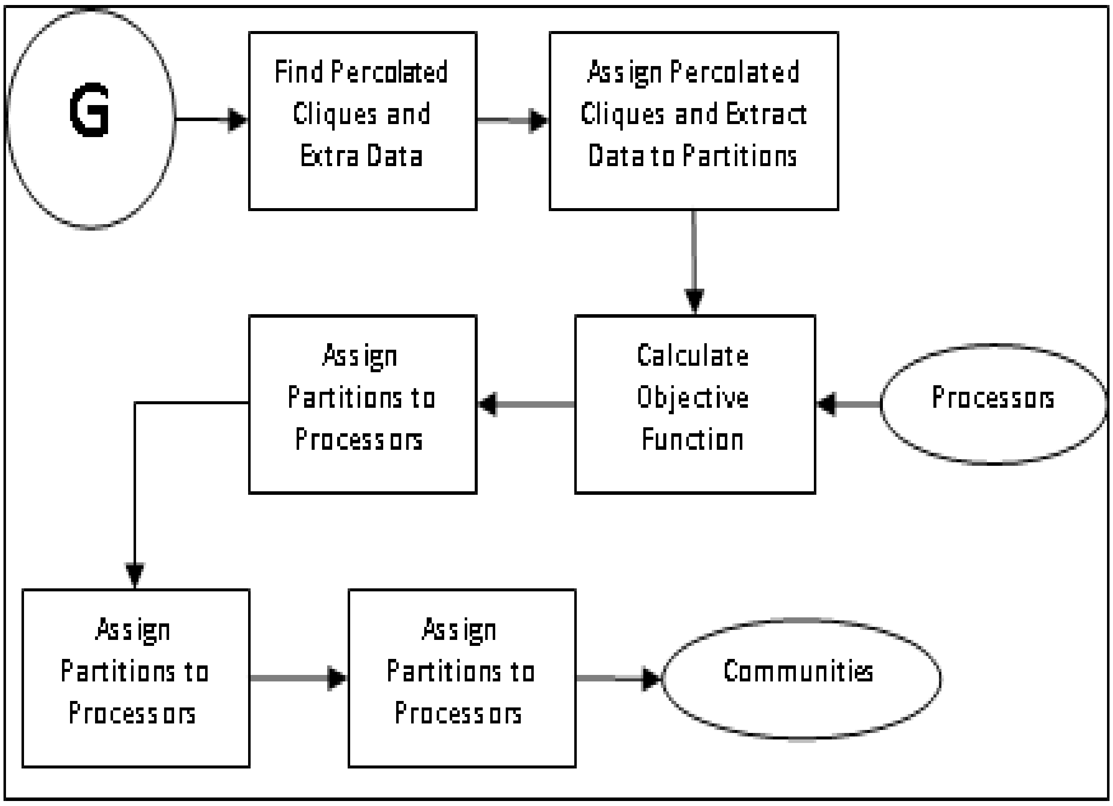

G
Find Percolated Cliques and Extra Data
Assign Percolated Cliques and Extract Data to Partitions
Calculate Objective Function
Processors
Assign Partitions to Processors
Assign Partitions to Processors
Assign Partitions to Processors
Communities

Figure 3.7 Influence Propagation Model

- Third, this model is expanded to identify denser areas where communication frequency is high. Inter community interactions are monitored.
- Fourth, Verifying the effectiveness and efficiency of this approach against the performance metrics decided.

This approach is appropriate for dynamic graphs for prediction of activities of communities. The most active nodes are identified which affect other nodes adjacent to them and help in expanding the communities by adding new members. Most active members are prone to become most influential members of the community. Thus their activities are monitored.

This method is applicable in various domains of applications like, link prediction, advertisement floating on social networks etc.

It is evident from the above discussion that cliques have a significant application in SNA especially in community detection. Next section describes significant work done by the researchers to devise time efficient clique finding algorithms.

3.4 FINDING MAXIMUM CLIQUE

Clique is defined as the complete subgraph in a graph [12]. Finding cliques in a graph is a problem of graph. The most famous variant of this problem is finding clique in a graph having maximum number of nodes, called maximum clique problem. In the graph shown in figure 3.8 the maximum clique is of size 4 consisting vertex 2, 3, 4 and 5.

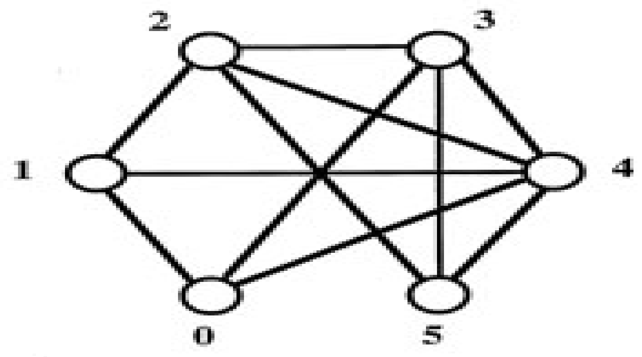


Figure 3.8 Undirected Random Graph

A wide range of applications of maximum clique in prominent fields are as community detection in networks [28, 29, 30], data mining in biometrics [140], information retrieval [141], data mining [142], disease classification based on symptoms correlation [143], computer vision [144], coding theory [145], and pattern recognition [146]. Some other applications of maximum cliques are listed in [147, 148]. Having applicable in a wid range of applications, the biggest challenge of finding maximum clique in a graph is an NP-Hard problem [24].

The most basic approach to find maximum clique in a graph is enumerating through all cliques present in the graph of every size and then selecting the largest one. But enumerating through every clique present in the graph requires huge amount of time which comes out to be infeasible for large graphs. Carraghan et al. [149] proposed a simple approach to find maximum clique and successfully reduced total number of enumerations significantly by pruning the branches of enumeration which are guaranteed to not lead to the optimal solution. The proposed algorithm enumerates maximum clique containing vertex v_i for every vertex. At any point of time if the sum of remaining number of vertices and the size of clique being enumerated through v_i

becomes smaller than the size of largest clique already found in enumeration through any other vertex, this branch of enumeration is immediately pruned.

Ostergard [150] proposed an algorithm for finding maximum clique in the graph by introducing an additional pruning step along with the pruning steps used in [149]. Ostergard algorithm performs faster than Carraghan algorithm on random benchmark graphs of DIMAX [151]. However, the pruning strategy is majorly affected by the order of processing the vertices.

Tomita et al. proposed an efficient branch and bound based MCQ algorithm [152, 153, 154] for finding maximum clique in a graph is based on vertex coloring. Vertex coloring is used to define the upper bound for maximum clique. MaxCliqueDyn [157] is an extension of MCQ algorithm, in which tighter upper bounds are used which incur more computation cost. These are applied on portions of search space. Another extension of MCQ is BBMC algorithm [156]. It computes bounds and graph transitions more efficiently by sorting the vertices in constant time using bit string.

Algorithms proposed by Tomita et al. [152 - 153], Bomze et al. [155], Segendo et al. [156] and Konc and Janezic [158] are based on branch and bound approach. Prosser [158] presented a comparative study of exact algorithms for finding maximum clique.

The above mentioned efforts of researchers made in the area of maximum clique finding in a graph majorly focus on reducing the computational overhead of the process. Community detection is a significant area of application of maximal cliques. A time efficient clique finding algorithm will subsequently improve time required for community detection in large social graphs.

3.5 SUMMARY

This chapter has presented a detailed study of literature related to various aspects of SNA. Data collection from social networks is the first requirement of SNA. Significant work done in data collection from social networks has been studied extensively. The challenges identified in the study of significant social network crawling strategies are gigantic size of the social networks and biasing in sampling algorithms. Algorithms focusing on elimination of biasing like NBRW, MHRW, DBSCAN, FS etc. are discussed in detail with their performance and limitations.

Later in this chapter, work done in the direction of community detection has been discussed. The study starts by examining existing techniques for community detection in which communities are considered disjoint and the social network is considered static in nature. But the social networks are dynamic in nature and therefore, an evolution in communities can be seen across multiple snapshots of the social network. Community detection techniques focusing on evolution pattern of the communities in dynamic social networks are studied further. In reality, the communities in social graphs are not disjoint but they are overlapping. A huge amount of work has been done to detect overlapping structure of communities using k-cliques. Significant frameworks using k-clique percolation have been discussed. In every clique based community detection methodology, finding clique in the network is a prerequisite step which is also computationally a costly affair. Existing algorithms to find cliques (especially maximal cliques) in the complex and large graphs have been discussed in last section of this chapter.

The challenges identified by the study of the literature are huge size of the network, biasing in sampling of the social network, dynamic behavior of social networks, high computation cost required in community detection and in finding maximal cliques in a graph. These challenges are addressed in proposed frameworks discussed in upcoming chapters.

CHAPTER 4

CRAWLER & FRESHNESS MAINTAINER FOR SOCIAL NETWORKS AND INTERACTION NETWORK BUILDER FOR NON-SOCIAL DOMAINS

4.1 INTRODUCTION

Social network analysis has gained significant importance in various domains of researches such as social research, market research and analytical projects, just to list a few [46 - 62]. Chapter 3 presented an in-depth detail about the vast applicability, required tools and the barriers before social network analysis. On the basis of literature studied in detail during initial phase of this research work, few objectives have been identified in the light of prominent applications of social network analysis and are being achieved in this research work. Primary objective is to design a social web data collector and freshness maintainer, which can collect data not only from non-social web domains as well as from trivial social networks giving it structure of social networks. In addition, the freshness maintainer is expected to maintain the freshness of the dataset of dynamic social network. Another objective is to design a tool for community detection in the network, which finds cliques and maximum clique to detect communities in social network.

To achieve the above stated objectives already mentioned in detail chapter 1, this research uniquely contributes a three-phased framework for social network data collection and analysis. The abstract view of the proposed framework is shown in figure 4.1.

The first objective pertaining to data collection from social networks and maintaining the freshness of the sample collected for dynamic social networks is being accomplished during first two phases. Owing to gigantic size and the dynamic behavior of social networks, the data collection and analysis of social network are two primary challenges.

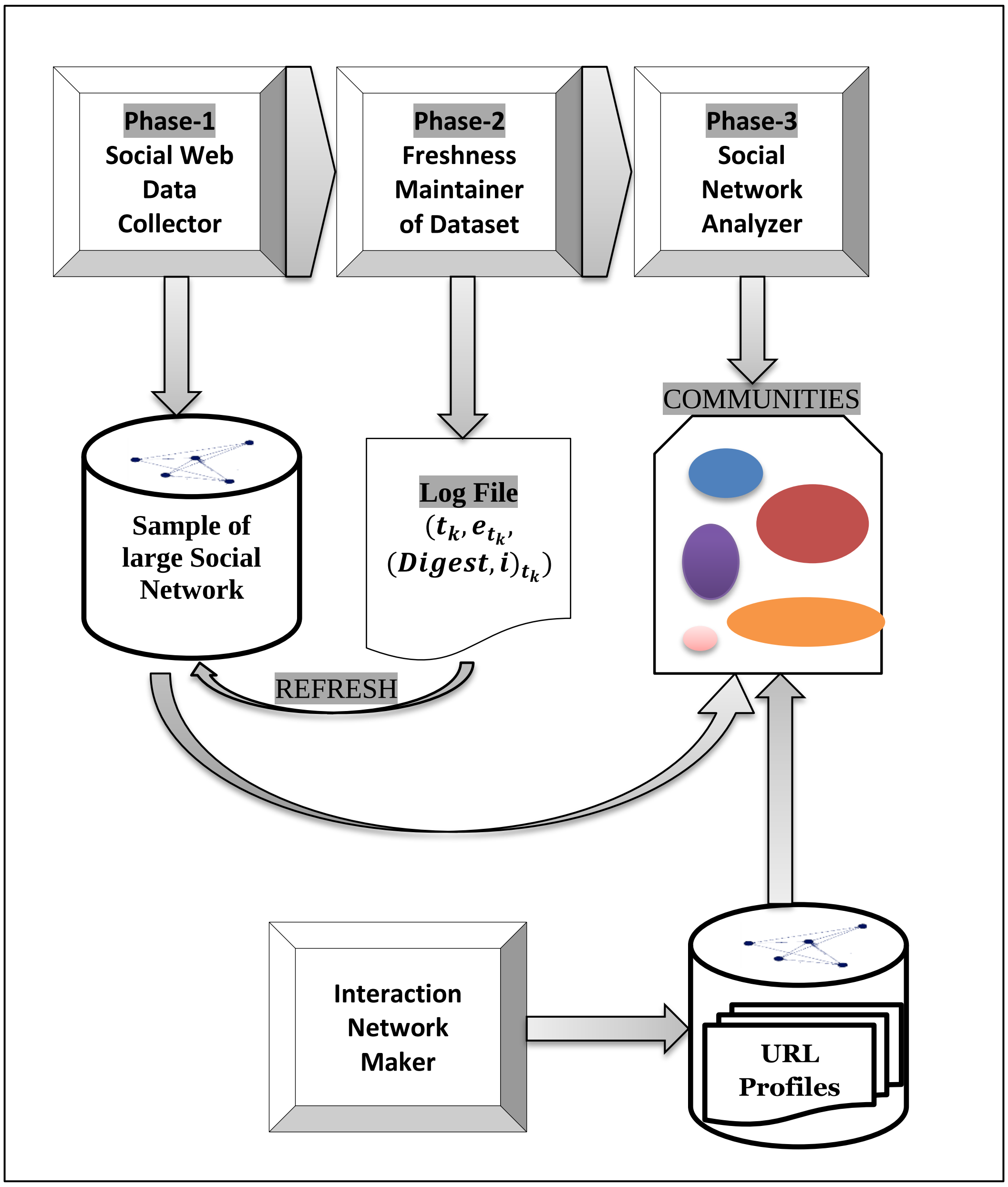

Phase-1
Social Web Data Collector
Phase-2
Freshness Maintainer of Dataset
Phase-3
Social Network Analyzer
COMMUNITIES
Sample of large Social Network
Log File
$(t_k, e_{t_k}, (Digest, i)_{t_k})$
REFRESH
Interaction Network Maker
URL Profiles

Figure 4.1 Abstract View of the Proposed Framework

Hence, in order to address the data collection issue, a representative sample of the social network is being collected instead of collecting the data of entire social network. The process of data collection faces another challenge known as ‘biasing’ [40, 65] i.e. the crawler gets biased towards similar kind of nodes. In order to address the above stated barrier, Phase 1 proposes a crawler named as *Cluster Coverage Sampling Crawler*, henceforth would be referred as CCSC [41]. CCSC considers clustered structure of the social network and offers a solution to biasing and is a natural representative of the already clustered social network.

The dynamic behavior of a social networks leads to data which might not prove to be relevant in current context and time. The challenge is handled in phase 2 of the proposed work i.e. Freshness Maintainer of Dataset. CCSC is extended to behave as *Foot Prints Based Crawler* (FPBC) [42] for social network, which handles dynamic behavior of the social network. FPBC leaves its foot prints in a log file during crawling of the social web. These foot prints are re-traversed by the proposed crawler to ensure freshness of the sample collected in previous crawl.

The proposed framework also contributes an *Interaction Network Maker*, which is responsible for creating a social network from WWW which are not inherently social networks but can be expressed as social networks. In particular, *Topical Interaction Network Builder* (TINB) [43] successfully creates a topical social network of web pages from WWW. This topical social network can be treated in the same fashion as any other trivial social network.

Community detection is a significant aspect of social network analysis and is widely used in various fields of research. Phase 3 proposes a time efficient *K-Clique Based Community Detection using Union Find* (KCUF) [44] algorithm which employs a subroutine named as *Maximum Clique Finder* (MCF) [45] algorithm. KCUF is a community detection framework based on k-cliques. It also finds maximum clique in any graph using MCF algorithm. In fact, finding maximum clique is a NP-Hard problem [24] problem. MCF uses several optimization pruning strategies to ensure that clique finding is time efficient. KCUF is responsible for detecting communities in the social graph. KCUF uses MCF for finding cliques and makes use of Union Find to merge similar communities into one. Detailed description of phase 3 has been discussed in chapter 5.

This chapter presents detailed description encompassing first objective of the proposed framework i.e. data collection from trivial social networks and maintaining the freshness of the sample data, collected from dynamic social networks and realizing non-social domains as social networks.

4.2 THE CLUSTER COVERAGE SAMPLING CRAWLER (CCSC)

Usually, social web crawling is similar to conventional web crawling [12] but these are different in many aspects [43]. In customary web crawling, to initiate crawling, the crawler is fed with the list of URLs. These URLs are called seed URLs. Crawler picks URLs one by one and fetches the page referred by the seed URL and in turn identifies new URLs on this page. URLs extracted from crawled page are filtered by eliminating already visited URLs and new URLs are added to the list of URLs to be crawled continually. Social web crawler also works in the similar fashion but it has different information structure. The structure of social web consists of nodes and their associations. Here, nodes represent users of the social web. In social web crawling, nodes have to be crawled instead of web pages. Hence, the crawler targets nodes as well as their association. Information in the form of attributes of the node can also be collected.

Generally most of the social networks are hidden behind login page. In order to traverse a social web, crawler need login credentials of seed node. Crawler enters in social web from this seed node. This node is also put in the list of nodes to be crawled. Crawler picks a node from the list of nodes to be crawled (for first time seed node is picked). It also fetches profile of the picked user and explores friend list of the picked node to find its associations with other nodes on the social web. The nodes extracted from friend list of the picked node are filtered to eliminate already visited nodes and new nodes are added in the list of nodes to be crawled. The information of connectivity among the nodes is maintained using graph representation approach, like adjacency matrix or adjacency list or edge list. While visiting a node on social network, a crawler also collects other information of the node such as Home Town, Study Place, Political Orientation etc. in the form of attributes of the node crawled [67, 75, 80]. Crawler iteratively keeps on picking nodes from the list of nodes to be crawled. Edge list and the attribute values are produced as the end result of the crawling process.

As mentioned already, the gigantic size of social networks is a big challenge for social web crawlers. To deal with this situation, crawling every node of the social web is not a good choice. Instead, sampling is being advocated in the literature [75]. Sampling favours that instead of gathering all the nodes of a complete network, few representative nodes shall be collected as sample of the social web. The sample of the social web is of benefit if and only if it has similar structural properties as does the original social network. The quality of sample collected from original social network depends on the quality of algorithm used for sampling the nodes.

There exist numerous sampling algorithms [63, 64, 70, 71, 73, 74, 75, 76, 77] but biasing is a common challenge in almost every sampling algorithm. The social web crawler gets easily attracted towards similar kind of nodes, especially towards the node with higher degree. The sample collected from the original social network may not have same structure as the original network because there is probability that crawler gets interested in similar kind of nodes and ignores other significant nodes. In this case the sample usually does not have similar structural properties as the original network. Therefore, biasing has to be handled efficiently.

On the similar grounds, another hidden challenge prominent in social networks is that people tend to collaborate or connect with people having something in common, like their interests, political orientations, same study place, same home town, same work place etc. These are called attributes of the nodes. Therefore almost every social network can scale indefinitely in any direction i.e. the network becomes denser at some places and sparse at other places. Nodes in a particular denser area of social network have common values in more attributes as compared to the nodes of other denser area. These denser portions can be considered as clusters and people/actors in these denser portions exhibit some kind of similar characteristics. Moreover, social network can be thought of as a collection of overlapping clusters (few clusters can be standalone also) where the overlapping area of clusters has nodes having similarities with nodes in both clusters. These nodes act like a bridge between two clusters.

This research considers a social network which can be represented as a social graph $G = (V, E)$, where V is collection of nodes and E is collection of edges representing associations among nodes. Due to scale-free nature of social graphs, the graph can be considered to have several overlapping clusters $CL_1, CL_2, CL_3 \dots \dots CL_k$ such that $G = \bigcup_{1 \le i \le k} CL_i$. There is possibility that few of these clusters may not be overlapping or are completely disjoints. Such clusters have least significance in social graph and can be ignored. Here, clusters can be thought of as form of communities or less restricted communities. Moreover, there is a greater possibility that each community that is excavated from sample of the graph definitely has a parent cluster. Let graph $G_s = (V_s, E_s)$ be the sample of graph G. $Co_1, Co_2, Co_3 \dots . Co_m$ be the communities detected in G_s, such that $G_s = \bigcup_{1 \le j \le m} Co_j$. Then, the predicate shown in equation 4.1 is always true.

$$\forall_j [Co_j \rightarrow \exists_i [CL_i \; is \; parent \; of \; Co_j]], where, 1 \leq j \leq m \; and \; 1 \leq i \leq k \quad (4.1)$$

Here, the proposed CCSC focuses on the above stated fact with assumption that there is no disjoint cluster in the social graph. The reason for this assumption is that, if the seed node from which the crawler enters into the social network lies in a standalone cluster, then nodes from that cluster can only be crawled. The crawler does not move out of that cluster and explore more nodes; therefore, the seed should be chosen very carefully so that no self-trap situation appears for the crawler. The crawler starts with seed node and begins exploring its friends for sampling. Most of the friends of any node belong to the same cluster, to which this node belongs. Nodes which are in overlapping area of two or more clusters have friends belonging to different clusters. After having sampled sufficient number of nodes from one cluster, crawler must proceed towards such nodes which are in overlapping area of two or more clusters. Once crawler encounters nodes in overlapping area, it gradually shifts from one cluster to another. The sample collected in this way reflects almost exact overlapping clustered view of original network only if the above predicate shown in equation 4.1 holds.

Based on the above discussion, following conclusions can be drawn:

(i) *Structure of almost every significant social network is clustered. Major portion of the complete social network has overlapping clusters.*

(ii) *Clusters can be categorized on the basis of specific attribute values of the nodes belonging to that cluster.*

(iii) *Nodes in overlapping area have additional values of their attributes which are specific to some other clusters.*

(iv) *A good quality sample of any social network exhibits the same clustered view of the original social network in terms of structural properties of social graph, such as degree distribution, modularity etc.*

Above mentioned conclusions led to CCSC. CCSC overcomes most common challenges of sampling algorithms to produce a good quality sample. It employs an adaptive sampling algorithm called Cluster Coverage Sampling (CCS) algorithm [41], which autonomously takes decision based on the attribute values of sampled nodes. CCSC holds certain conventions listed as follows:

(i) *CCSC assumes that the associations in targeted social network do not have directions, which means the social graph produced by this crawler will be an undirected graph.*

(ii) *CCSC is interested in publically available information of the social network. Several social networks have privacy features for their users which enable users to customize the availability of their personal information and to decide the audience for the information. Users may also wish to make few details publically viewable. CCSC is interested in information which is made public by the user.*

(iii) *CCSC assumes that the social network is fully connected. This implies that CCSC deliberately ignores standalone clusters.*

Figure 4.2 depicts the architecture of CCSC. The crawler module is fed with the seed node and its login credentials where seed node belongs to any cluster of the network. Further, it is to be noted that the seed node must not belong to some standalone cluster. Though there is no suitable method to check this fact but the initial attribute values given to seed node play a vital role in this. Most of the time instead of creating a user on social network for CCSC, an existing user is used as seed node, so that the crawler can proceed further by migrating over friends of the seed node.

CCSC gets activated when the seed node S_n is given as argument to the Crawler function. Credentials of S_n are used by crawler to enter in Social network. CCSC maintains a data structure C_n that contains the list of nodes to be crawled. CCSC logs in social network as seed node S_n. Therefore, the only node crawler has access to is S_n. Hence, $\mathrm{S_n}$ is the first node added to $C_n[\,]$. In addition, CCSC maintains two more datasets namely, *Sparse_Data_Set* and *Sample_Data_Se*t where, *Sparse_Data_Set* contains every node that is to be explored by the crawler from the latest sampled node. Once CCSC visits a node *v*, it explores its friends list. If the friends list of visited node *v* is not locked, the nodes in friends list of *v* are collected by CCSC. At this point CCSC observes about the associations between node *v* and nodes in its friend list. These associations are stored in *Sparse_Data_Set*. There is a strong possibility that these nodes have connections among themselves too. But these associations are unknown until CCSC visits every node.

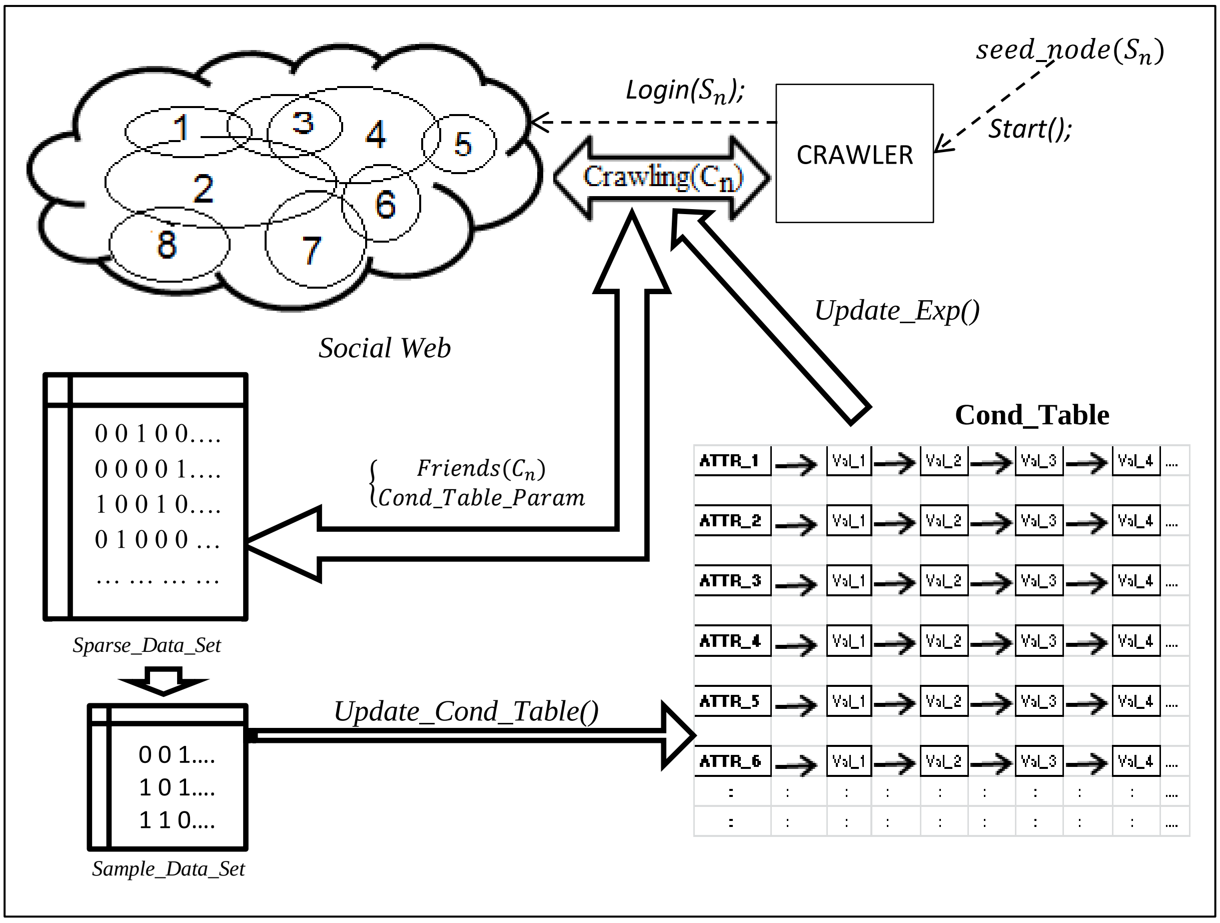


Figure 4.2 The Proposed CCSC Framework

It can now be concluded that information about the associations of only those nodes is now known which are visited by CCSC. All the collected nodes are in the list to be crawled further. These explored nodes compete for getting sampled in next iteration. Every node explored in previous iteration, should not be visited by crawler as the target is to collect a sample instead of collecting complete social network. Out of many explored nodes, few are selected (sampled) for further exploration. Only those nodes are selected which fulfill the sampling criterion (Sampling criteria is decided by the sampling algorithm used by the crawler). In case of CCSC, Cluster Coverage Sampling (CCS) algorithm is used and the sampling criterion defines the target cluster and is defined in terms of Boolean Expression which is generated by the Expression Generator Algorithm (discussed later in this chapter). The sampled nodes are visited further and information about their associations with other nodes is known whereas for the nodes which do not qualify the sampling criteria, their association information is not known. The sample is considered of good quality only if it has similar structural characteristics, as does by the original social network. Keeping this fact in view, the competing nodes which do not qualify sampling criteria must be discarded from the dataset along with their associations. Hence, a filtered dataset is also maintained called as Sample_Data_Set which contains only those nodes, whose complete associativity is known. These nodes are now visited by the crawler; therefore, attribute values of these nodes are also known and are stored in the data set. Algorithm 4.1(a) and Algorithm 4.1(b) depicts the working of CCSC.

```
Algorithm- Crawler
Crawler($S_n$)
    Input: Seed_Node $S_n$.
Start-
    Perform login with credentials of $S_n$ ;
    Initialize Sample_Date_Set and
Sparse_Data_Set;
    $C_n[\,] \leftarrow S_n$;
    Crawling($C_n[\,]$);
End
```

Algorithm 4.1(a) : Crawler Algorithm

Crawling function makes use of sophisticated datasets such as *E_Nodes, Competing_Nodes*, *Cond_Table* and *EXP* that are maintained to govern the crawling process throughout. The function in focus is the major routine in CCSC and it runs continually until the termination condition is met where the termination condition is likely to be unreachable as it demands the sampling of entire social network i.e. there are no new nodes to be crawled. Hence any criteria such as *x* percent of associated nodes (friends) of P_n or Competing_Nodes can also be set for termination of sampling. The current work considers *10%* of friends of P_n as the termination condition.

The proposed Crawling algorithm represents actual crawling process where list of sampled nodes $C_n[\,]$ is fed to crawling process. Nodes contained in $C_n[\,]$ are visited by the crawler one by one so that these nodes are stored in Sample_Data_Set. CCSC extracts one node say P_n from $C_n[\,]$. Profile of P_n is visited by CCSC. It extracts targeted information from the profile of P_n.

```
Algorithm- Crawling
Crawling($C_n$[ ])
   Input: Nodes_To_Crawl_List $C_n$[ ].
Start-
   Update (Sample_Date_Set, $C_n$[ ], Sparse_Data_Set);
   While($C_n$[ ] ! $empty$)
   {
        $P_n$ ← Select_&_Remove($C_n$[ ]) ;   //$P_n$ is current node being processed.
        E_Nodes ← Find_Friends($P_n$);
        Competing_Nodes ←Find_Redundency(E_Nodes, id_Table);
       Update (Sparse_Data_Set, Competing_Nodes); //nodes explored and their
connections are saved.
        Cond_Table ← Fetch_Attr_Values ($P_n$);
   }
   EXP ← Expression_Generator(Cond_Table);
   $C_n$[ ]  ← Sample_Nodes (Competing_Nodes, EXP);
   Crawling($C_n$[ ]); //Recursive call.
End
```

Algorithm 4.1 (b) Crawling Algorithm

Firstly, CCSC extracts friend list of P_n, which contains nodes directly connected to it. These nodes and their association information are stored in *Sparse_Data_Set*. There is a strong possibility that the friend list of P_n contains one or more nodes which have already been explored by the crawler and need not be stored in dataset. These duplicate nodes need to be handled at this stage. For this purpose a temporary list *E_Nodes* is created. Direct friends of the node P_n get stored in *E_Nodes* temporarily for redundancy check. To check the redundancy another data structure is maintained i.e. id_Table, which is a direct access table hashed upon unique IDs of nodes. Almost every online social network assigns each user a unique identification number called ID of that node. To check redundancy of the nodes during crawling, this unique ID of node is checked. When CCSC explores a new node for the first time, its entry is made in *id_Table* using its unique ID. When a node is encountered again by the crawler, its entry is already present in *id_Table* and the node is discarded from future considerations. The nodes in *E_Nodes* go through this redundancy check. To make it fast, *id_Table* is maintained using hashing. In case of duplicate nodes, collision occurs. But there is a possibility that two different nodes having different IDs may collide in *id_Table*. In such cases it is harder to decide whether it is a duplicate node or a collision for two different keys. To resolve this dispute unique IDs of the nodes are stored at the position on which the ID is mapped by hashing function. Once the collision occurs the IDs can be compared to decide whether to make new entry in the table or discard. Chaining has been used for collision resolution. All the nodes in E_Nodes are checked for redundancy and new nodes explored by the crawler are added to another list called *Competing_Nodes*. *Competing_Nodes* now becomes the list of nodes which are unique and are currently competing to get sampled. CCSC samples nodes from *Competing_Nodes* for further iterations.

The second information of P_n collected by CCSC is values of its attributes. Generally, users in almost every social network have certain attributes like '*Lives in', 'Work', 'Study Places'* etc. Users fill values in these attributes as per their choice and interests. It is highly probable that people get associated with others if they have common values in atleast one of the attributes. This theory also explains the denser area (cluster) and sparse area of the social network. Nodes falling in denser area are prone to have common values for more attributes as compared to that of nodes falling in sparse areas. CCSC collects these attributes and their values. A data structure, *Cond_Table*, is maintained by the crawler which contains attributes of users (nodes) and their possible values as shown in figure 4.2. The objective is to collect every possible value of an attribute in the Cond_Table encountered so far. For example, there are several values possible for an attribute *'Lives in'* because different users live at different places and one user might have lived at several places. *Cond_Table* stores every value as set of possible values for attribute '*Live in*'. It helps in finding the overlapping region of the clusters. Suppose there is a user *u*, who had lived at two places *p1* and *p2*. *u* might be part of two clusters, one cluster of people from place *p1* and other cluster of people from place *p2*. *Cond_Table* is continuously refreshed as crawling proceeds and new nodes are visited by the crawler. When crawler visits a new node, attributes of that node (if found any new) and their possible values (if found any new) are updated in *Cond_Table*.

CCSC targets to reach overlapping area of the cluster after collecting sufficient nodes from one cluster in order to change the cluster. Proceeding towards overlapping area and then to other cluster means CCSC is selecting nodes corresponding to their attributes and their values. The values of attributes play a major role in shifting crawler from one cluster to another. The idea is to label a cluster with an expression of attributes and their values. This implies that, as per the attribute values of nodes belonging to one cluster, a boolean expression can be created in such way that it is TRUE for every node in that cluster. The boolean expression is the definition of a particular cluster, therefore, a definition of the cluster is created in the form of Boolean expression and nodes satisfying this definition are selected in the sample created by the crawler. Now crawler needs to change the definition of the cluster and the cluster automatically changes. CCSC uses attributes and their values to create boolean expression, *EXP*. Nodes which satisfy *EXP* are sampled next.

EXP contains several attributes with their one possible value ANDed or ORed, thereby forming a nested boolean expression. CCSC uses an autonomous algorithm for expression generation. *Expression_Generator* algorithm is depicted in algorithm 4.2.

```
Algorithm-Expression_Generator
Expression_Generator(Cond_Table)
Start-
  k=0;
  while(k<Num_Of_Attributes(Cond_Table))
  {
    $Atom_k$ ← Select_Value($ATTR_k$);
    $On\_Atom\_Op_k$ ← Select_Op(On_Atom_Op_List);
    $Between\_Atom\_Op_k$ ← Select_Op(Between_Atom_Op_List);
    Update(EXP, $On\_Atom\_Op_k$, $Atom_k$, $Between\_Atom\_Op_k$);
  }
  Return EXP;
End
```

Algorithm 4.2: Expression Generator Algorithm

Following are some conclusions based on above discussion:

(i) *Cond_Table is used as main ingredient of expression EXP.*

(ii) *There are k attributes in Cond_Table, explored so far by CCSC.*

(iii) *In the expression EXP, there will be k Atoms, one for each attribute.*

(iv) *One combination of attribute and its value is considered as an 'Atom' (*$Atom_k$ *can be defined as* $\left(ATTR_k == Value_j(ATTR_k)\right), 1 \leq j \leq total\ num\ of\ possible\ values\ of\ ATTR_k$*).*

(v) *AND and OR are binary operations which require tow operands, therefore these are used between operands and hence put in a set of operations called Between_Atom_Op_List.*

(vi) *NOT is unary operator used on any Atom optionally and it is put in On_Atom_Op_List.*

(vii) *To make expression generation process completely autonomous and random, Values of attributes, binary operators Between_Atom_Operator and unary operator On_Atom_Operator are selected randomly.*

CCSC uses EXP as the definition of the cluster. The nodes in set *Competing_Nodes* are considered for sampling by CCSC. Not every node from *Competing_Nodes* will make it to the sample and there is no fixed pattern by which the nodes from *Competing_Nodes* are sampled. CCSC tries to sample nodes randomly yet wishes to avoid any kind of biasing and *EXP* solves the purpose. CCSC limits the number of nodes to be sampled for one cluster by declaring a threshold value (τ). For every cluster, CCSC tries to sample nodes up to. Threshold for sampling (from one cluster) has been decided on the basis of two factors: number of nodes in *Competing_Nodes* and previous pattern. Previous pattern implies that the size of current cluster is probably similar to the previously encountered clusters. Sample Nodes algorithm is shown in Algorithm 4.4.

The sample nodes are selected randomly from *Competing_Nodes*. Moreover, nodes are selected one by one and only those nodes which satisfy *EXP*, are sampled from *Competing_Nodes*. There is a possibility that there are not enough nodes in *Competing_Nodes* which satisfy *EXP*. In this case to achieve τ number of nodes sampled in this iteration of CCSC, nodes are selected randomly and *EXP* is avoided to make sample unbiased. If the crawler could not find τ amount of nodes in any particular iteration of CCSC, the threshold for the next iteration is reduced. In this way the history of threshold values affect the future thresholds.

The above proposed framework CCSC mainly focuses two objectives viz. (1) handling biasing and (2) covering clustered structure of social network. CCSC maintains a Boolean expression *EXP* of attributes of users of social network and their values.

```
Algorithm- Sample Nodes
Sample_Nodes (Competing_Nodes, EXP)
Start-
   s_count = 0; r_count=0; //To count number of sampled nodes and randomly
selected nodes
   while(s_count < Threshold(Competing_Nodes) &
r_count<SizeOf(Competing_Nodes))
   {
      r_count++;
      c ← RandomSelect_&_Remove (Competing_Nodes); //c is current node being
processed
      If( c holds EXP)
      {
         s_count++;
         add(Node_List,  c);
      }
   }
   while(s_count < Threshold(Competing_Nodes))
   {
      Add(Node_List, RandomSelect_&_Remove(Competing_Nodes));
      s_count++;
   }
   Append(id_Table, Node_List);
   Return Node_List;
End
```

Algorithm 4.3: Sample Nodes

EXP is the definition of the type of nodes CCSC currently wants to crawl. The attributes and their values are almost similar for most of the nodes in a cluster in social network. Hence, nodes satisfying *EXP* are more probable to belong to the same cluster. CCSC changes the *EXP* to have nodes of different flavour time to time. Nodes having different attribute values belong to different cluster in the social network and therefore, with the change in *EXP*, the cluster in social network also changes. This strategy helps CCSC to cover clustered structure of the social network in the sample collected and as the *EXP* keeps changing throughout the crawling process; crawler may never get stuck to one type of node. The biasing is also resolved. The proposed algorithm is tested on a synthetic social network and the results are discussed in chapter 6. Experimental results show that the samples collected using CCSC are highly effective and unbiased.

A good quality sample is absolute representation of complete social network and its properties on a smaller scale. Unbiased sampling and coverage of clusters results in collection of good quality sample but it is not enough. The quality of sample may downgrade with time due to dynamic behaviour of social networks. In case of dynamic social network, the quality of sample gradually downgrades with time and a good quality sample may get obsolete after some time. Therefore, social network dynamism is a serious issue and must be handled in appropriate manner. The next section extends CCSC with abilities to detect temporal changes in dynamic social networks.

4.3 FOOT PRINTS BASED CRAWLER (FPBC) FOR DYNAMIC SOCIAL NETWORKS

Social networks having the ability to change their structure with time due to change in associations among actors or change in users' interests and orientations etc. are popularly known as Dynamic Social Networks (DSN) [12, 26, 73, 78, 128, 129] and it is worth mentioning that most of the online social networks are dynamic in nature. Social network dynamism is of two types, one in which the nodes create new associations as well can leave old associations whereas, the other one allows nodes to make new associations but already existing associations are not removed. For example, facebook is an example of former kind. On facebook, users are allowed to add new friends in the friend list and can also unfriend the existing friends, whereas Live Journal is an example of later kind of dynamism. Live Journal is a social network of researchers in which two researchers are connected if they have worked together on some research article. Here any researcher can collaborate with any researcher; he/she hasn't worked with. This adds a new association in the network but no option is available for the researchers to disassociate with other researchers. The quality of the sample depreciates with time in such cases. Temporal changes of the dynamic social networks must be detected and reflected in the sample to retain samples validity. The current work embodies around a novel technique to handle such dynamic behaviour of the social network. The objective of the proposed technique is to detect the changes occurred in the actual social network with respect to the latest sample collected from it. Idea is to trace the social network on the same path followed by the crawler in last crawl. When crawler samples the social network, it keeps creating footprints on the path so that the same path can be traversed again in future if required. The proposed methodology would be henceforth named as Foot Prints Based Crawler (FPBC) for dynamic social network.

As already discussed in section 4.2, CCSC is unbiased and is capable of generating clustered view of the social network. The main idea is to keep changing the direction of the crawler time to time by autonomously changing *EXP*. Temporal changes are frequent in DSN with respect to the sample collected by crawler and are required to be identified. These changes can easily be identified by traversing the same path followed by the crawler in previous iteration, again. In CCSC, the direction of the crawler or the nodes crawler is going to crawl next is highly affected by two factors i.e. Boolean expression fed to the crawler and current seed node (first node sampled after change in the Boolean expression). This is the initial point from where the crawler starts sampling nodes after having changed the definition of targeted cluster. Therefore, it can be stated that the crawler can leave its footprints in the form of this definition of crawler and initial point for this definition. Such footprints, if recorded properly, can be used for re-traversing the same path. The footprints of the crawler must be recorded every time the crawler changes its direction, which means whenever the definition of the targeted cluster is changed the crawler must leave its footprints.

This section proposes FPBC, a crawling framework to handle graph dynamism and is basically an extension of CCSC. FPBC collects a fresh sample from the social web CCS algorithm as illustrated in CCSC. Now, while the collected sample is a good quality sample but it gets obsolete in case of DSN. To handle the dynamism of social web, FPBC has been equipped with ability to leave foot prints in an iteration of crawl and follow these foot prints to check freshness of the sample in next iteration of crawl. The very first crawl of FPBC is to collect the sample from social web and it is termed as iteration 0 of FPBC. In iteration 0, FPBC collects sample in the same manner as done by CCSC. But in addition to this, FPBC also records its foot prints in a log file.

Other iterations (> 0) of FPBC have been executed to refresh the sample collected in iteration 0. Foot prints are recorded in a log file called *Foot_Print*. The structure of *Foot_Print* log file is shown in Table 4.1.

Table 4.1 : Structure of Foot_Print Log File

Time $\boldsymbol{t_k}$	**Expression $\boldsymbol{e_{t_k}}$**	**Digest $(\boldsymbol{Digest, i})_{\boldsymbol{t_k}}$**
Timestamp, log file is indexed over timestamp.	Expression generated at time $\mathrm{t_k}$	Digest created at time $\mathrm{t_k}$ in i^th iteration of crawling

The foot print is created in first iteration *(i = 0)* of crawling. In later iterations *(i > 0)* the digest has been compared with this base digest to verify freshness of the sample in data repository. The crawler starts with seed node O_{t_0}. Log is initialized with default foot print created at timestamp $\mathrm{t_0}$ which contains O_{t_0} in digest and expression is NULL. Crawler enters in network with credentials of seed node O_{t_0}. Crawler initializes list of nodes to be crawled, C, with O_{t_0}. Function *Crawl_Freshness* ($C, O_{t_{0,}}$, i) calls a subroutine *Crawl_Fresh_Detect(C, i)* which is main routine for the crawling process. It is shown in Algorithm 4.4. C is the list of nodes to be crawled next and i is the iteration number of the crawler. When *i* is *0*, FPBC is collecting a fresh sample and do not need to do verification of freshness.

```
Algorithm: Crawl_Freshness($C, O_{t_{0,}}$, i)
/*C is set of nodes to be crawled before sampling other nodes C is initially empty*/
/*$\boldsymbol{O_{t_0}}$ is seed node from where crawler starts in each iteration*/
/*i denotes iteration number*/
begin-
            Perform login with credentials of $\boldsymbol{O_{t_0}}$ ;
            Initialize Sample_Data; /*Repository to stored crawled graph*/
            k=1;/* next log will be created with time stamp $\mathrm{t_k}$*/
              C ← C U $\boldsymbol{O_{t_0}}$;
              Crawl_Fresh_Detect(C, i); /*i^th iteration of crawler*/
end
```

Algorithm 4.4: Crawl_Freshness Routine

To detect the freshness of sample an algorithm *Crawl_Fresh_Detect* has been proposed as shown in algorithm 4.4. Each node v_n of C is selected one by one and their neighbours are explored. *N(v_n)* contains neighbours of v_n. *Sample_Data* is the repository in which nodes and their association are stored. *Sample_Data* is updated with the neighbours explored so far. This process is repeated until C has no more nodes to explore further. Then the digest of *Sample_Data* collected so far is created. The digest is tagged with timestamp and iteration number of the crawler. If it is first iteration of the crawler then the log is created. If $i>0$ then the freshness of the sample is verified by comparing latest digest with the digest present in log for previous iteration i. e. $i-1$.

It is least probable that the latest digest and previous digest are exactly same. There will be a deviation between them. A tolerance limit is determined for deviation of latest digest from base digest. The tolerance limit is decided on the basis of type of social network. As already stated in previous section DSN have been classified to be trivial and non-trivial social networks. Trivial social networks such as facebook provide privilege to its actors to form new associations and to remove existing associations. While non-trivial social networks such as LiveJournal, the authors are allowed to form new connections but disassociation of previously associated authors is not present or extremely exceptional. So the tolerance limit in both the cases will be different. As the trivial social networks are very prone to change the tolerance to change should be higher as compared to the tolerance to change for social networks from other domains such as LiveJournal. Up to a tolerance limit the sample is considered to be fresh enough to sustain inferences derived from it. If the tolerance limit is crossed, the sample in the repository is considered obsolete and it is reported by the crawler. In this situation the base sample is discarded and the fresh sample is gathered and stored in the repository. This process is generalized for any dynamic social network. Algorithm 4.5 depicts the working of *Crawl_Fresh_Detect().*

FPBC has been implemented as a layered architecture for crawling dynamic social network and maintaining freshness of sample dataset.

```
Algorithm : Crawl_Fresh_Detect(C, i)
/*C is set of nodes to be crawled*/
/*i is iteration of crawler*/
begin-
    while $C \neq \phi$ do
            $v_n$ ← C\$v_n$;/*$v_n$ is selected and removed from C*/
            N($v_n$) ← Find_Friends($v_n$);/*N($v_n$) contains neighbours of $v_n$*/
            N ← N U N($v_n$);/*N contains neighbours of all nodes crawled so far*/
            Update(Sample_Data, N); /nodes explored and their connections are
        saved*/
    end
    Generate_digest($(Digest, i)_{t_k}$, Sample_Data);
    if $i=0$ do
            Create_Log($t_k, e_{t_k}, (Digest, i)_{t_k}$);
            k++;
            EXP ← Generate_Expression(Attributes, values);
    else
            Verify_Freshness($(Digest, i)_{t_k}, (Digest, 0)_{t_k}$);/*subroutine to test
freshness
                of sample*/
            if $(Digest, i)_{t_k}$ is fresh enough do
                    EXP ← Fetch_Log($e_{t_k}$);
            else    Report ERROR and return;
            end
    end
    C ← Sample nodes from N which satisfy EXP;
    Crawl_Fresh_Detect(C, i); /*Recursive call*/
End
```

Algorithm 4.5: Crawl_Fresh_Detect Algorithm

The framework is shown in figure 4.4. In the architecture two kinds of iterations are shown. For i = 0, FPBC has various layers of functionalities. Each layer is a module responsible for some specific task described as follows:

(i) *First layer of FPBC, i.e. Login module, establishes the connection between the framework and social web with valid credentials of seed node O_{t_0} to login.*

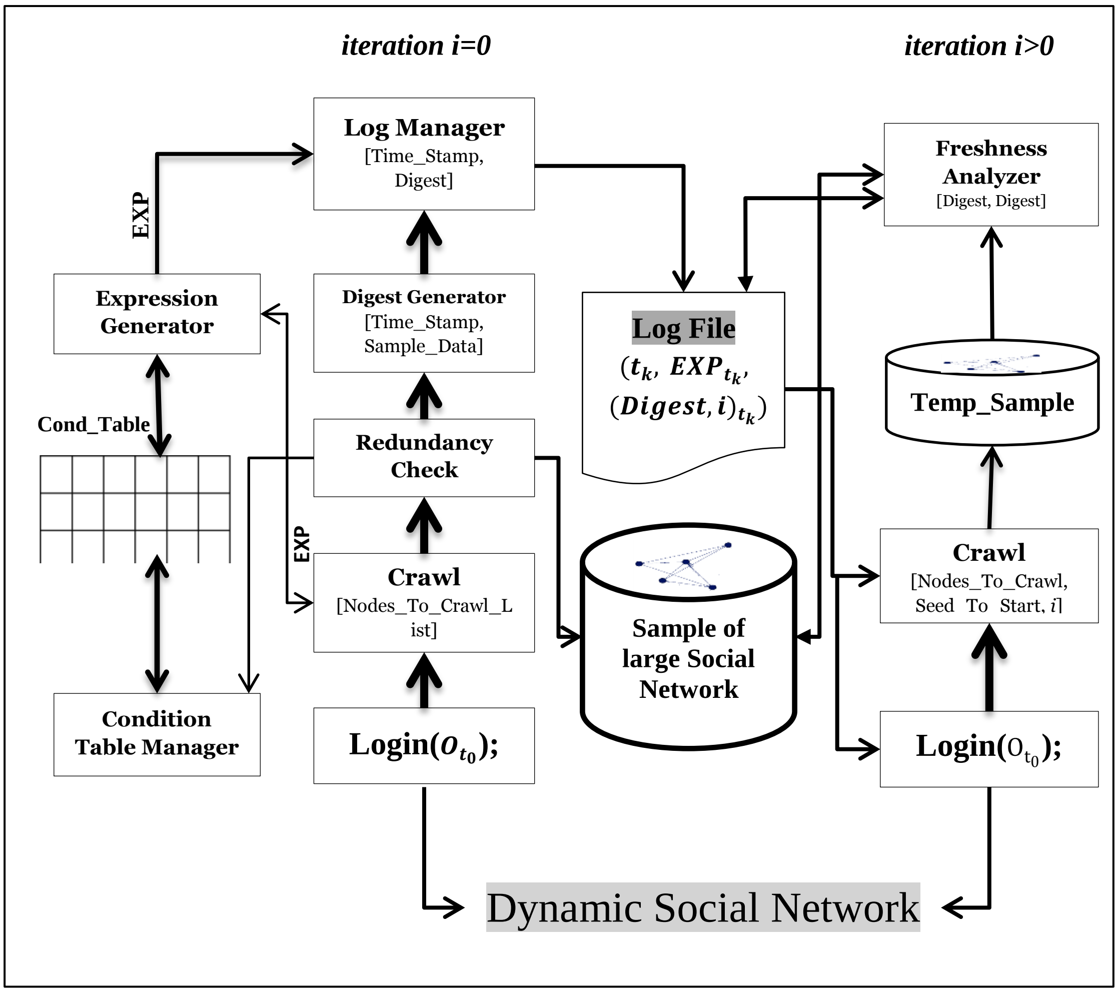


Figure 4.3 Foot Prints Based Crawler (FPBC)

(ii) *Next layer is Crawl module. It is responsible to collect nodes, associations among nodes and their attributes from the social network.*

(iii) *Expression generator is a module responsible for generating boolean expressions. Expression Generator module uses Cond_Table for expression generation. Cond_Table is maintained by Condition Table Manager module. Crawl layer selects only those nodes which satisfy the expression EXP generated by Expression Generator.*

(iv) *The sample collected by Crawl layer is stored in a repository after redundancy check. Redundancy Check layer is responsible for elimination of duplicate nodes and associations. Redundancy Check layer makes use of id_Table which is hashed upon unique IDs of nodes, as used in CCSC.*

(v) *Next layer is digest generator layer which creates digest of the sample dataset collected so far. The digest generally contains structural blue print of the sample. For instance, degree distribution in the sample, modularity etc. represent structural characteristics of the sample.*

(vi) *The digest along with expression is stored in a repository called Log File. Log File is maintained by a layer called Log manager layer. The layer is responsible to record foot prints of FPBC with time stamp.*

(vii) *After complete execution of all layers for iteration 0 (base iteration) of FPBC, the datasets created are – sample dataset, Log File, and Cond_Table.*

FPBC collects a sample in base iteration using cluster coverage sampling. The sample is unbiased and depicts clustered view of the social network. After certain period of time, when it is assumed that the freshness of the sample may have expired, FPBC can be run repeatedly to verify and maintain freshness of the sample stored in repository. When FPBC is run for the purpose of freshness verification and assurance, the iteration is greater than 0 $(i > 0)$. For iteration $i > 0$, FPBC functions in following manner:

(i) *The crawler uses same seed node to log in into social network which was used in first iteration. This seed is obtained from Log File created in base iteration of FPBC.*

(ii) *The Crawl layer is responsible for running extensive crawling process of FPBC. Iteration $i > 0$ of FPBC is different from first iteration in following ways: (1) Now crawler does not run autonomously, instead, it uses log file to proceed further. (2) Crawler does not bother to collect attributes and their values to update condition table, because, (3) No expression is generated in this iteration and the expressions logged in previous iteration are used. (4) FPBC only collects nodes satisfying the expression and generates a temporary sample.*

(iii) *Freshness Analyzer layer is responsible to determine freshness of the sample. Freshness analyzer layer compares digest of temporary sample and* *with the digest of base sample created at the same time stamp.* $(Digest, i)_{t_k}$ *is compared with* $(Digest, 0)_{t_k}$.

(iv) *The Deviation between structural properties of current sample and previous digest is measured in terms of error. The sample is considered obsolete if the error is more than predefined threshold value (tolerance). This verification has been also done by Freshness Analyzer layer.*

(v) *If the sample is found obsolete at time stamp* t_k, *the sample dataset is updated from this point onwards and the current crawler iteration is set to i = 0.*

FPBC successfully identifies freshness factor of the sample and if the sample is found obsolete, a fresh sample is collected from that point. FPBC also detects the pattern of temporal changes in dynamic social network. The duration, in which the sample gets obsolete, can be determined by the analysis of previous iterations of FPBC. For instance, after time *T*, the second iteration *(i = 1)* FPBC has been run to check freshness of the sample collected in previous base iteration. If in this iteration *(i = 1)* the sample is found up to date, then the next iteration *(i = 2)* of FPBC will be run after *T*(1/F)* time where *F* is the freshness factor of the sample. *F* is proportional to the deviation of current sample from base sample.

FPBC successfully generates the clustered view of the social network deploying CCS algorithm and is also able to determine the freshness of the sample in the local dataset. To determine the freshness, FPBC runs in two iterations. In first iteration, FPBC collects original sample of the social network. While collecting sample FPBC also maintains a log in which the path followed by crawler and compressed foot print of the sample called digest is maintained. In second iteration, FPBC does not run autonomously, instead it follows its foot prints of previous iteration. The digest in foot prints is compared with the sample being collected in second iteration. If the deviation is within tolerance limit then the sample is fresh otherwise the sample is obsolete and needs to be updated. FPBC is also able to identify the pace of dynamism of any social network.

It is now an understood fact that crawling is a lengthy and rigorous task which is sometimes fruitless because of some directions being useless. Usefulness of the direction of crawling is subject to the application domain. For instance, if the crawling is being done for search engines like Google, then every direction is useful as users' queries are not known in advance. But if the purpose of crawling is to gather data from web related to a specific topic or domain then any web page not matching to the target context is fruitless and must be avoided by the crawler. At this point the crawler needs decision power to examine all the directions and decide the most appropriate direction to proceed further. The crawler should have the capability to categorize the web page as relevant or irrelevant to the focus of the crawler. The categorization conditions are defined in terms of predefined parameters. The parameters are the contextual information related to the topic (focus) of the crawler. Every possibility is checked with the given parameters. Such crawler is called parameterized crawler. Although parameterized crawlers have been in existence [82 - 93] but very few researchers have focused towards building a focused crawler for social networks [79]. No pure focused crawler for a structured data w.r.t social web from WWW which is inherently a non-social network is present in literature. The upcoming section proposes *Topical Interaction Network Builder (TINB),* which is a parameterized crawler for building topical interaction network from www. Unlike social network, normal web has different structure and therefore a different procedure for crawling has been proposed in this research.

4.4 TOPICAL INTERACTION NETWORK BUILDER (TINB)

Normal web crawler starts at a seed URL. It visits the web page of seed URL and extracts all the URLs present at this page for further crawling. Instead of adding every URL found at the parent page to the frontier the crawler selects only those URLs which satisfy the certain set of parameters.

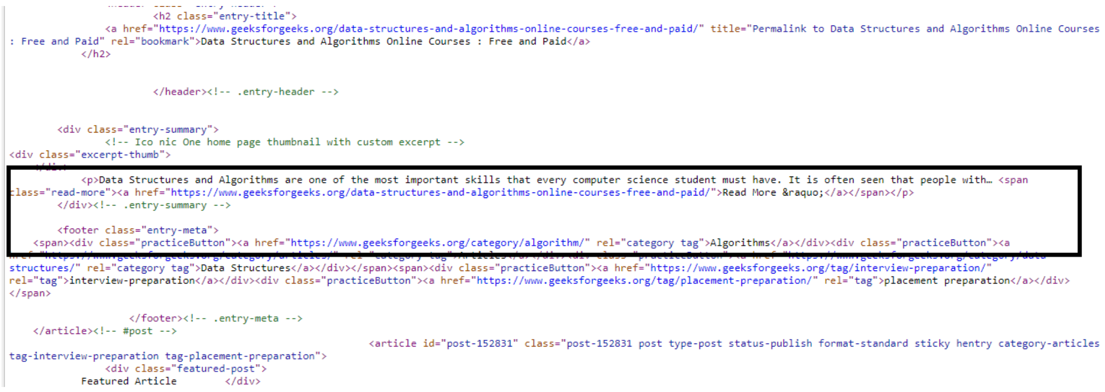
Predefined parameters are used to create a boolean expression. Boolean expression is the definition of the focus of crawler. The keywords related to the focus of the crawler with their positive or negative orientation are used as truth values in the expression. These truth values are ANDed and ORed to create an appropriate definition of the focus of the crawler. Every URL is tested with this expression and only those URLs whose contexts satisfy the expression are visited by the crawler.

The context of a link is the text present around the links on the web page [85, 86]. One of the early attempts confines the context to the anchor text only [87] i.e. the URL and the text used within the anchor tag is considered as context for the URL. As shown in figure 4.4, the text written between anchor tag (*Click for best anchor tag example.*) and the link (*https://www.demoanchortag.com*) is considered as context for URL. Only this information is tokenized to generate keywords, which are passed into Boolean expression for determining the focus.

```
<a href=”https://www.demoanchortag.com”>Click for best anchor tag example.</a>
```

Figure 4.4 Anchor Tag Example

Another developed approach used anchor text and some the text in its vicinity (or text window) [88] as context for the URL. A text window is decided and captured within which the target URL lies. As shown in figure 4.5, the text coming in the vicinity of the window is considered as context for the URL.



```
                    <h2 class="entry-title">
                <a href="https://www.geeksforgeeks.org/data-structures-and-algorithms-online-courses-free-and-paid/" title="Permalink to Data Structures and Algorithms Online Courses
: Free and Paid" rel="bookmark">Data Structures and Algorithms Online Courses : Free and Paid</a>
            </h2>


                    </header><!-- .entry-header -->


        <div class="entry-summary">
                <!-- Ico nic One home page thumbnail with custom excerpt -->
<div class="excerpt-thumb">
            <p>Data Structures and Algorithms are one of the most important skills that every computer science student must have. It is often seen that people with… <span
class="read-more"><a href="https://www.geeksforgeeks.org/data-structures-and-algorithms-online-courses-free-and-paid/">Read More &raquo;</a></span></p>
        </div><!-- .entry-summary -->

        <footer class="entry-meta">
    <span><div class="practiceButton"><a href="https://www.geeksforgeeks.org/category/algorithm/" rel="category tag">Algorithms</a></div><div class="practiceButton"><a
href="https://www.geeksforgeeks.org/category/articles/" rel="category tag">Articles</a></div><div class="practiceButton"><a href="https://www.geeksforgeeks.org/category/data-
structures/" rel="category tag">Data Structures</a></div></span><span><div class="practiceButton"><a href="https://www.geeksforgeeks.org/tag/interview-preparation/"
rel="tag">interview-preparation</a></div><div class="practiceButton"><a href="https://www.geeksforgeeks.org/tag/placement-preparation/" rel="tag">placement preparation</a></div>
</span>

                </footer><!-- .entry-meta -->
    </article><!-- #post -->
                                                            <article id="post-152831" class="post-152831 post type-post status-publish format-standard sticky hentry category-articles
tag-interview-preparation tag-placement-preparation">
                <div class="featured-post">
            Featured Article        </div>
```

Figure 4.5 Text Window Example

The text in the vicinity of text window is now considered as context for the URL. It is processed as follows:

i. *This text window is parsed and tokenized to find significant keywords. The text may contain several insignificant words such as ‘is’, ‘the’, ‘a’, ‘an’, ‘on’, ‘to’ etc. It may also contain adjectives and adverbs, which are insignificant to determine the focus area. Such tokens are ignored during identification of keywords.*

ii. Every other URL in this text window is ignored. Only the target URL and other text are considered. The other URLs are treated independently, when they are about to be visited by TINB.

iii. The keywords obtained in the text window are categorized in different categories corresponding to the Boolean expression, it has to satisfy. The Boolean expression defines the focus area of TINB. Boolean expression contains contextual information of the focus area defined in terms of keywords. If similar keywords are discovered around the target URL then this URL is related to the focus area. The keywords obtained from parsing are then tested against the Boolean expression. Attributes in Boolean expression are defined next.

iv. The expression has attributes like domain_of_URL, key_in_URL, Key_in_left_of_URL, Key_in_right_of_URL and Key_in_anchor.

v. Attribute domain_of_URL maps target domains of the URLs. Mostly the domain of the URL is informative to the topic to which the web site is related to. For instance domain .ac.in is generally used for university web sites. Therefore domain of the URL is considered as one of the significant attributes.

vi. Attribute key_in_URL attribute identifies the URL against the focused area. For instance if our focus area is 'Travel' then keywords like outing, journey, hotel, bus, train etc. in URL imply higher probability of such URLs being related to our focused area.

vii. The text window is captured around URL. Text appearing to the left of the URL is called left context of the URL and the text appearing to the right of the URL is called right context of the URL.

viii. In various situations relevance of the URL may be more affected by left context or may be more affected by right context of the URL. It depends on the focus area and the web page. Therefore, if it is not sure that which of left or right context has more effect on relevance of the URL, both should be given same importance.

ix. Attribute key_in_anchor contains only those keywords which are within anchor tag. This is sometimes as relevant as the key_in_URL keywords or sometimes it may contain some more trivial text like 'click here' etc.

The focus area for the crawler is defined in terms of values of these attributes. The values of attributes can be predefined as per the focus area. An expression is generated with these attributes, their predefined parameters and logical operators (AND, OR, NOT). The expression is generated in fully disjunctive normal form (OR of ANDs), which means that each attribute appears in every conjunction of the expression. Every URL extracted from web pages has its own set of keywords for each attribute of the expression. These keywords for the attributes are tested against every conjunction of the expression. The URL is assigned a score based on the number of conjunctions satisfied by the URL. The score portrays the probability of the target URL being relevant to the focus area. Relevance probability of a URL u is defined as shown in equation 4.2.

$$P(u) = \frac{Number\ of\ conjunctions\ satisfied\ by\ u}{Total\ number\ of\ conjunctions} \tag{4.2}$$

A threshold value can be set for the acceptance of the probability. Any URL having relevance probability equal to or greater than the threshold value is considered relevant and it is added to the URL frontier to be crawler further. The threshold for relevance probability depends on the focus area and availability of web pages in that focus area.

The strategy for focused crawling has been devised in this section which will be used by TINB during crawling and building interaction network. Purpose of TINB is not just to crawl www and collect data in datasets but TINB is dedicated to convert this data into a social network of web pages. This social network of web pages is called interaction network of web pages which is in all respect similar to any traditional social network. This implies that every web page in interaction network is same as a user in a traditional social network. The interaction network must be exactly similar to any traditional social network so that all SNA techniques are applicable on interaction network without any modification. To achieve this, every web page is treated as a user and it is given a profile as a user of social network. This is called profiling of web pages which is explained in next section.

4.4.1 Realising Normal Web as Social Web (Profiling the URLs)

In any trivial social network the users or actors have a general profile. The profile contains basic information of that user as attributes of the user such as home town, study place, work place, interests, political orientation etc. The users have ties between them which are realised as directed or undirected edges depending upon the kind of associations. The associations among users are highly affected by the attributes of the users. It has been already discussed that how values of these attributes motivate interactive association among users. All SNA techniques make use of this profile oriented structure of the social network. SNA techniques use associations among users and their profile information for attribute values to derive conclusions.

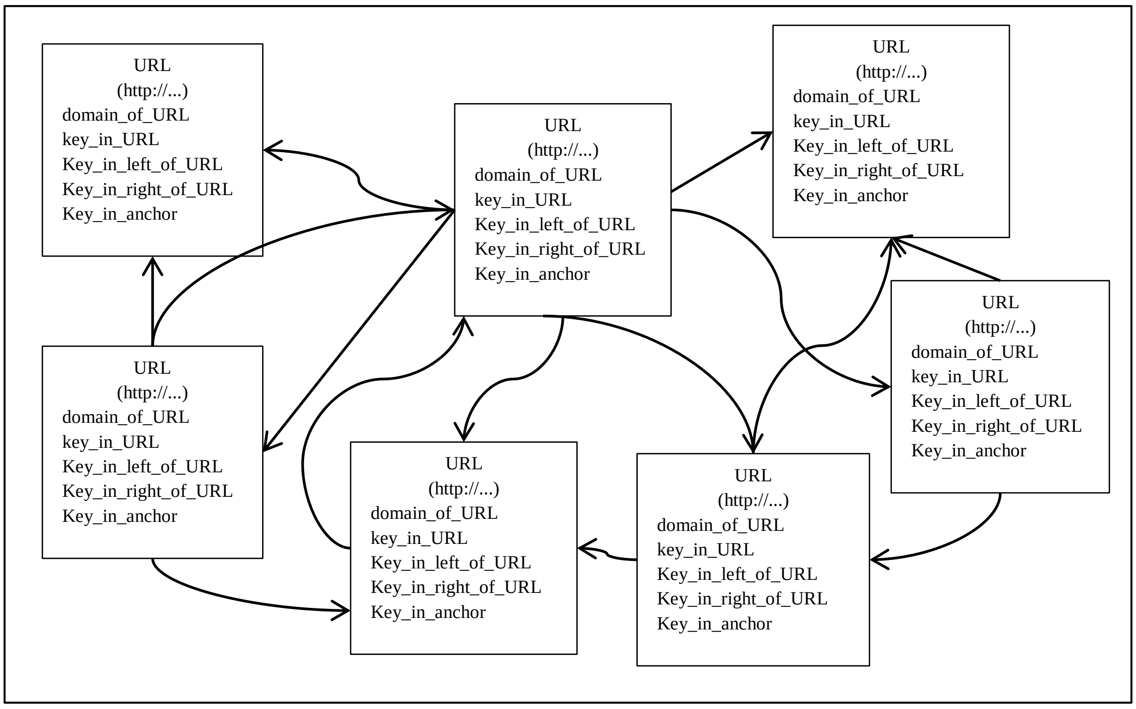


Figure 4.6 Prototype of Interaction Network of Normal Web with URL Profiling

The interaction network prepared from non-social domain must also have same structure if SNA techniques are to be applied on that. Interaction network prepared from WWW shall be eligible for SNA techniques to be applied on it without any modification in the technique only if the interaction network has same structure as does by any traditional social network. Therefore, in realization of normal web as social network, URLs are portrayed as users. Each URL has its own profile. URL profile contains general information of the URL in the form of its attributes such as *domain_of_URL*, *key_in_URL*, *Key_in_left_of_URL*, *Key_in_right_of_URL* and *Key_in_anchor*. Figure 4.6 shows a prototype of the interaction network of normal web.

The interaction network of normal web contains exactly same structure of associations as well as profile oriented arrangement of URLs. Every SNA technique can be applied on this interaction network without any modification in the technique. This step is called profiling of URLs. In the next section complete architecture of TINB crawler has been explained in detail.

4.4.2 Topical Interaction Network Builder (TINB)

As shown in figure 4.7, TINB is a multithreaded four layered crawler. TINB is designed to be multithreaded to achieve distributed computing. Each thread has a sequence of four layers namely, Fetch layer, Processing Layer, Filter Layer, Processing Layer and Representation Layer. Every layer has specific task to complete. The functioning of layers is as follows:

A) Fetch Layer

The lowest layer is Fetch layer. It is responsible for establishing the connection with internet. URL frontier contains the URLs to be fetched further with their relevance probability (Equation 4.2). Redundancy has been handled and URL frontier contains non-redundant URLs only. Initially, seed URLs are stored in URL frontier. URL frontier is maintained by Filter layer. URL frontier is a shared resource. It is shared among threads. Fetch layer selects a URL from frontier which has highest relevance probability. The relevance is measurement of relevance of the URL to the topic of focus. The fetched page is stored in a temporary repository for further processing.

B) Processing Layer

Processing layer is the most responsible layer and it performs the following tasks:

i. *It parses the complete web page and identifies URLS on the page. These URLs are extracted from the page.*
ii. *The extracted URLs are checked for redundancy. Already visited URLs are discarded.*

iii. The text window is captured around the new URLs found on the page.

iv. Text window of every URL is parsed and tokenized.

v. The tokens are processed to eliminate unnecessary tokens. Stemming is performed on tokens to get the tokens in their root words e.g. 'likes' and 'liked' are converted into the root work 'like'. Significant tokens are designated to corresponding contexts of the URL.

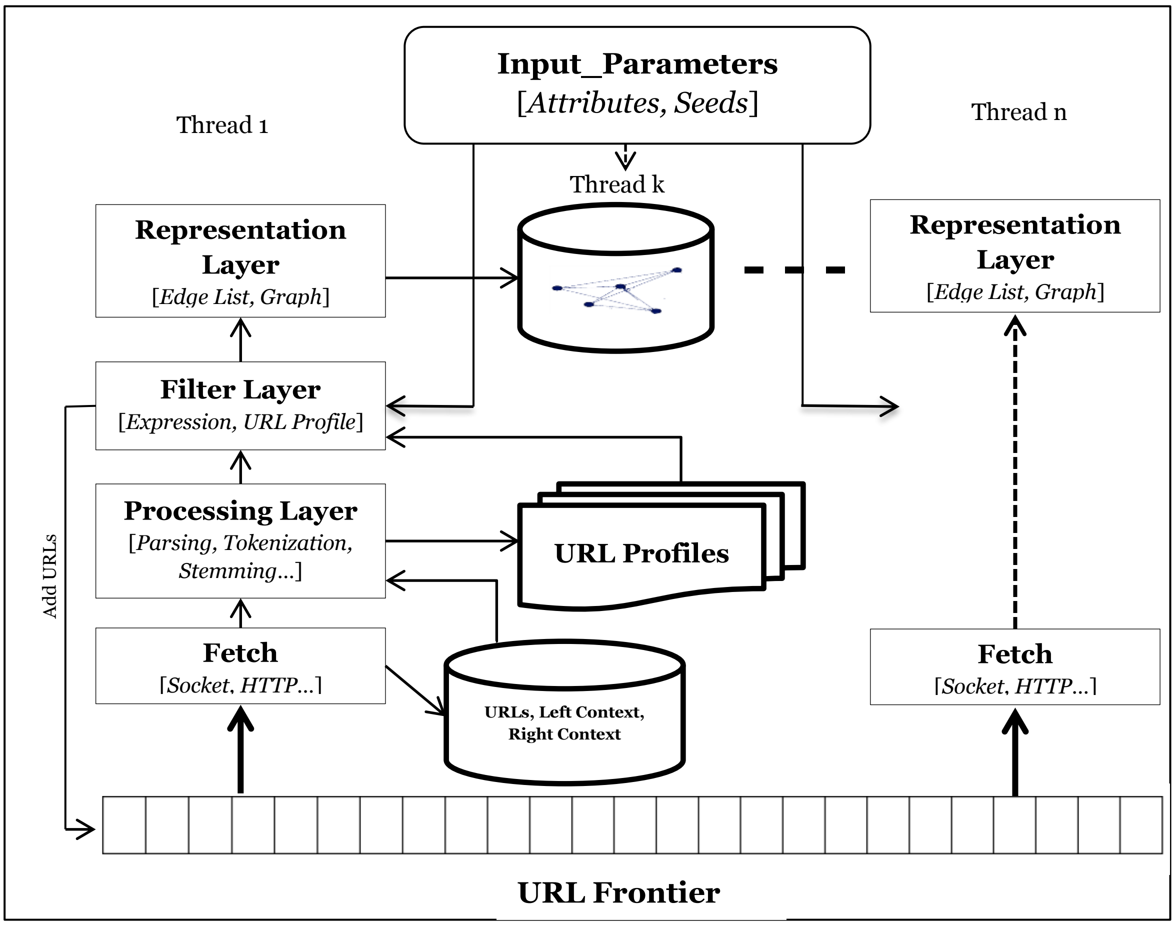


Figure 4.7 Multithreaded High Level model of Crawling Architecture

vi. The profile of each URL is prepared. The URL profile contains attributes domain_of_URL, key_in_URL, Key_in_left_of_URL, Key_in_right_of_URL and Key_in_anchor. The tokens found in text window are filled as values to these attributes. The association information is also maintained in the form of adjacency list. This adjacency list is considered as temporary because the URL profiles prepared in this layer will be evaluated for their relevance in next layer i.e. Filter layer. The URLs which are discarded in Filter layer will not be part of interaction network and therefore their associations with other

URLs will also be discarded. The prepared URL profiles are stored in profile repository, URL Profiles.

C) Filter Layer

URL profiles prepared by Processing layer are input to the filter layer. Filter layer is responsible for quantification of relevance of URLs to the focus topic. It performs following tasks:

1. *This layer takes predefined parameters for the attributes and the expression defined over attributes. The Boolean expression is the definition of focus topic. It is defined in terms of predefined values of attributes of URLs. The attributes and their predefined values are used to construct a Boolean expression. The expression has several conjunctions of attributes which are to be satisfied by the URL.*
2. *Each URL profile has values corresponding to its attributes in the form of contexts of URL. The values in attributes of URL and definition of focus area (Boolean expression) are matched.*
3. *Each URL is assigned a score called relevance probability on the basis of number of conjunctions satisfied by the URL (Equation 4.2).*
4. *Filter layer discards every URL having relevance probability below threshold value.*
5. *A threshold can be set for filtering out the URLs which seem to be irrelevant to the focus area before promoting them to next layer.*
6. *The URLs which survive filtering process are added to URL frontier.*

D) Representation Layer

Representation layer is responsible for realizing network of the normal web as social web. This network is similar to any other interaction network. It performs following tasks:

1. *It takes filtered URL profiles and from Filter layer and the temporary adjacency list prepared by Processing layer.*
2. *The nodes which have been rejected by the Filter layer are removed from adjacency list.*

iii. *Remaining associations are appended into central data set of interaction network.*
iv. *In central dataset the associations are maintained as edge list.*
v. *URLs with their profiles are users of the interaction network.*

The above architecture of TINB successfully crawls WWW and prepares a focussed interaction network. The interaction network is similar to any other traditional social network and hence any SNA technique is directly applicable on it.

4.5 SUMMARY

The chapter presented first two phases of the proposed framework and module for building interaction networks from nonsocial domains. The first phase proposed Cluster Coverage Sampling Crawler (CCSC) while second phase comprised of Foot Prints Based Crawler (FPBC). The chapter began by highlighting the major challenges pertaining to data collection from social networks and later provided a solution for data gathering from social networks with the objective of high quality fresh samples. Topical Interaction Network Builder(TINB) has been proposed as additional module in the proposed framework which is responsible for preparing social network of nonsocial domains. TINB could crawl WWW and could prepare an interaction network of web pages aligned with the specified topic. This interaction network is similar to any traditional social network. Therefore, every SNA technique can be applied on this interaction network.

Further, it is found that the sample is required to be analyzed and this can be done using any of the social network analysis techniques. Most effective and most widely used SNA technique is community detection in social graph. This requirement poses the need of next of phase of the current research work. Next chapter proposes a time efficient algorithm to find maximum clique namely Maximum Clique Finder (MCF) and a k-clique based community detection technique, KCUF.

CHAPTER 5

MAXIMUM CLIQUE FINDER (MCF) & K-CLIQUE BASED COMMUNITY DETECTION USING UNION FIND (KCUF): THE PROPOSED WORK

5.1 INTRODUCTION

As outlined in the previous chapter, the current research work is being carried out in three phases. First two phases of the current work has been discussed in chapter 4 while this chapter focuses on the last phase of the work. As already indicated, CCSC and FPBC crawlers could achieve the stated objectives partially. However; analysis of any social network involves discovering structural characteristics of social networks with respect to certain analytical objectives such as degree distribution, centrality measurements, clique identification, community detection etc. used in various applications such as [53 - 56, 59, 60, 62 159, 160, 161].

Literature advocates that social networks and social graphs can be used interchangeably [9] as social networks are conventionally represented using social graphs. A dataset pertaining to the social network is stored as social graph. Hence, every utility available for graphs can be applied on social graph representing the social network. Social Network Analysis uses several graph structural properties as tools for deriving analytical results such as degree distribution in the graph, modularity in the graph, centrality measurements, community detection etc. Every structural property of the graph denotes different aspect of the graph. The application in which social network analysis is done has different significance of every structural property of the social graph. For instance, if the most significant central node in the social graph is to be identified, then variations in centrality measurement can be used. It depends on the application for which this exercise is being done. Closeness centrality and betweenness centrality have different impact and different significance [9, 12]. Similarly, different structural properties shall have different significance in different applications. This chapter focuses on analysis of social network considering two structural characteristics i.e. clique identification and community detection, in particular. The third phase proposes an algorithm referred to as *Maximum Clique*

Finder (MCF) deployed in a novel algorithm namely *K-clique Based Community Detection Using Union Find* for finding cliques and community detection respectively. In fact, later is an extension of former. The two algorithms are being illustrated in following two sections.

5.2 RELATIONSHIP BETWEEN COMMUNITY AND CLIQUE

A community in a social graph is a group of nodes which are more closely connected to each other as compared to their connection to rest of the nodes in the graph. This implies that the interaction links among the nodes of a specific community are more as compared to communication links with other nodes that are not part of this community. Due to this reason community detection is one of the most used techniques for social network analysis. Simply stated, community represents a denser area in the graph [27]. A similar phenomenon is ‘clique’ [112]. A clique in a graph is defined as subgraph in which every node is connected to every other node which is part of the clique [112]. In other words, clique is a complete subgraph. As compared to a community, the nodes in a clique are more tightly coupled. The node in a community may not be directly connected to every other node in the community but most of its associates are part of the community. For instance, let’s consider a clique of size 50. Every node which is part of this clique should have degree atleast 49. Now let’s consider a community of size 50. Here, a node of degree 10 can also be part of this community, provided maximum of its 10 adjacent nodes are part of this community. This observation leads to following conclusions:

i. *A clique is definitely a community but a community may not be a clique.*
ii. *Density constraints in clique are higher as compared to community.*
iii. *If two cliques are adjacent to each other they can be merged in one community. Cliques of size k (k-cliques) are said to be adjacent if they have k-1 nodes in common.*
iv. *A clique of size k is said to be adjacent to a community of size n if they have k-1 nodes common. Such cliques can be merged into the community, they are adjacent to.*
v. *All adjacent k-cliques can be merged together to form a community of size n having [n – (k-1)] adjacent k-cliques.*

It is clear from the above mentioned observations that clique and communities are similar concepts with different level of density constraints and can easily complement each other with few implementation arrangements. Detecting communities in a graph using cliques involves two steps:

(1) Finding cliques in a graph, and

(2) Identifying and merging adjacent cliques to form bigger communities.

Literature indicates that community detection is not only a complex task but also is time consuming, when carried out on a social graph [95 - 99, 103, 104, 113 - 120]. There exist many algorithms for detecting communities and cliques in social network [131, 132, 134 - 139], however; these are possessed with one or the other shortcomings. For instance, the algorithms proposed in [132] and in [136] are accurate k-clique based community detection algorithms but do not perform well in terms of time.

The upcoming section proposes a time efficient clique finding algorithm which is then extended for community detection in the subsequent sections presented in this chapter.

5.3 MAXIMUM CLIQUE FINDER (MCF)

MCF is a time efficient community detection algorithm that detects communities in social network. As already mentioned, a clique is a complete sub-graph of a given undirected graph. Further, a graph may have multiple cliques of different sizes. Finding clique in a graph has multiple variants. Most popular variant of finding clique in an undirected graph is finding largest clique in a graph. Finding the clique that has maximum number of vertices in a graph is called maximum clique and this problem is called maximum clique problem of the graph. A simple brute force approach for finding maximum clique in an undirected graph is as follows:

Arbitrarily start with a vertex u and add it to a set C. C contains vertices which are part of the clique having vertex u. Initialize a set P, which is initially NULL.

Step 1: Explore adjacent vertices of newly inserted node in C and put them in set N.

Step 2: *Find P = P $\cap$ N. If P is empty go to step 6.*

Step 3: *Select one node v from P such that v is most probable node to be part of the maximum clique corresponding to node u, and insert it in set C.*

Step 4: *Go to step 2.*

Step 5: *Repeat step 1 to 5 for each vertex of the graph.*

Steps 1 to 6 enumerate through all the cliques corresponding to each vertex. The largest one can be chosen and it is called maximum clique of the graph.

The above algorithm is depicted as flow chart in figure 5.1.

Enumerating through every node is a lengthy task and incurs heavy time overheads. Therefore, finding maximum clique in a graph is an NP-Hard problem [24]. The major challenge in finding maximum clique in graph is its time complexity. The above mentioned brute force approach takes almost infeasible time for large graphs because it tries to enumerate every possibility. But in reality there are several possibilities which may not lead to a clique of size bigger than the size of already found largest clique. Such possibilities are considered fruitless. If the clique finding algorithm could determine the fruitlessness of such possibilities in early stages of exploration, the algorithm can ignore such possibilities. This strategy is called pruning [149, 150]. The unrewarding branches of computation are pruned and algorithm does not waste time in exploring such paths. In this way the time can be saved and finding maximum clique in a graph can be made faster. To do so the algorithm must have arrangements for identifying unproductive branches of computation. MCF exploits pruning at various steps in the process of finding maximum clique to improve time. MCF uses following notations:

1. *An undirected graph is denoted by G = (V, E), where V is set of vertices and E is set of edges.*
2. *The graph G (V, E) contains n vertices as $\{v_1, v_2, \dots\dots v_n\}$.*
3. *$Adj(v_i)$ is set of vertices adjacent to vertex v_i.*
4. *Cardinality of $Adj(v_i)$ i.e. degree of vertex v_i is denoted by $deg(v_i)$.*

v. Degree of each vertex is computed once in the beginning of the algorithm.

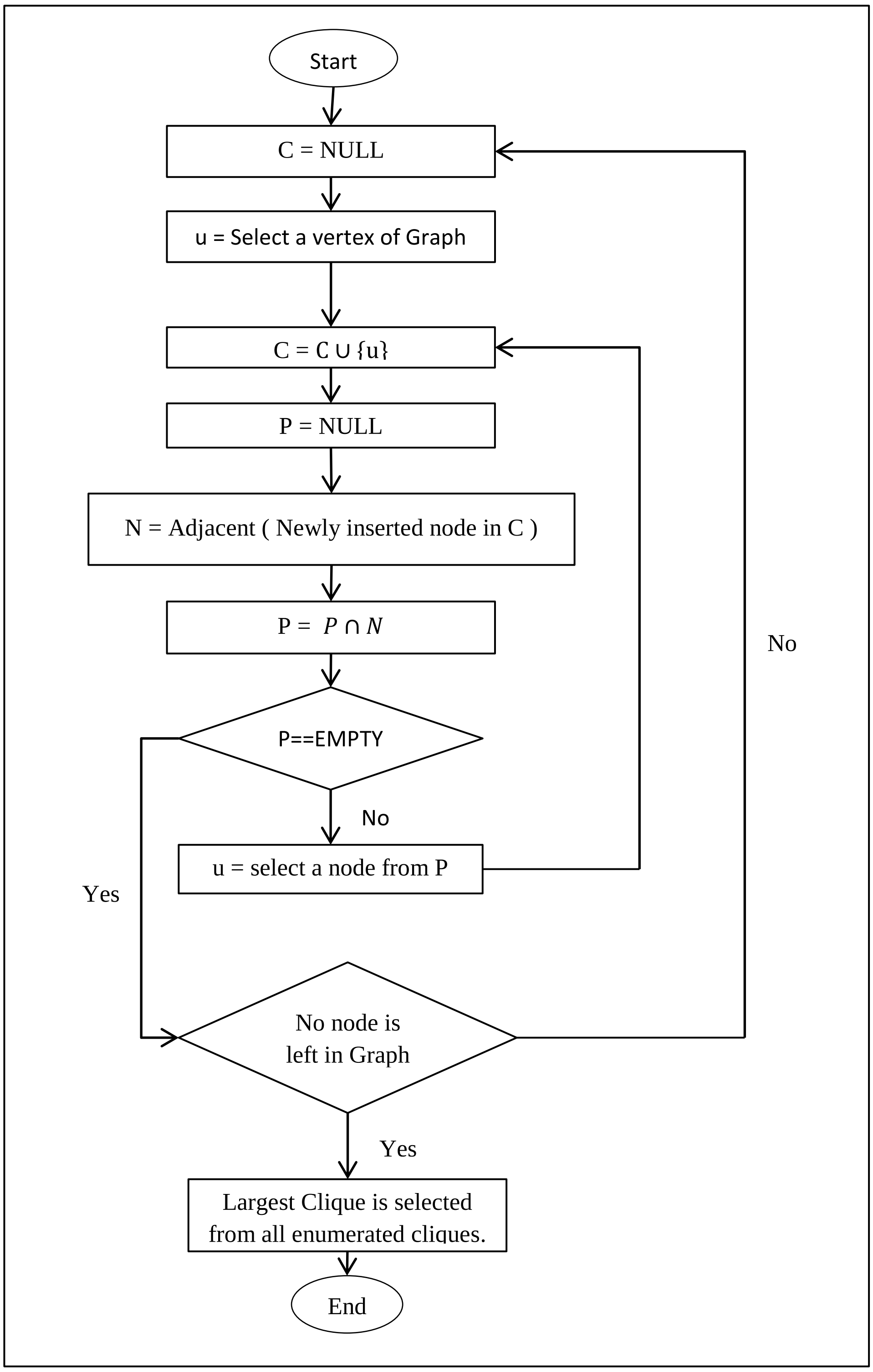

Start
C = NULL
u = Select a vertex of Graph
C = C ∪ {u}
P = NULL
N = Adjacent (Newly inserted node in C)
P = P ∩ N
P==EMPTY
No
u = select a node from P
Yes
No node is left in Graph
No
Yes
Largest Clique is selected from all enumerated cliques.
End

Figure 5.1 Maximum Clique Finding in a Graph

MCF implements four pruning steps to reduce search space considerably. The proposed MCF algorithm and the subroutine Find_Clique() are depicted in Algorithm 5.1 (a) and Algorithm 5.1 (b) respectively.

```
Algorithm : MCF($G=(V,E)$)
    /*max, size, C and M are global variables.*/
begin
    Sort all the vertices in V non-increasing order of degree./*pruning 1*/
    $max \leftarrow 0$;
    for $i \leftarrow 1$ to $n$ /*to iterate n times*/
        $size \leftarrow 1$;
        $C \leftarrow \phi$;
        $v_i \leftarrow$ Select_First($V$); /* $v_i$ is highest degree node in V.*/
        $V \leftarrow V \backslash \{v_i\}$; /*Pruning 2*/
        $current\_deg \leftarrow deg(v_i)$;
        $C \leftarrow C \cup \{v_i\}$; /*C is current clique containing $v_i$*/
        $U \leftarrow V \cap Adj(v_i)$;
        Find_Clique($U$,$1$,$C$);
        if $size > max$ then
            $max \leftarrow size$;
            $M \leftarrow C$; /*M is max-clique found so far*/
        end if
        if $max = current\_deg$ then /*Pruning 3*/
            return M;
        end if
    end for
    return M;
end
```

Algorithm 5.1 (a) Maximum Clique Finder (MCF)

The variable *max* stores size of the largest clique found so far. The pruning decision is based on the value of *max*; as it contains size of the largest clique know till now. Initially it is set as 0 because the graph has not been explored yet. But it is not necessary to initialize *max* with 0 always, instead, *max* can be initialized with any lower bound if cliques of size smaller than the lower bound, are insignificant.

Maximum clique containing a vertex v_i cannot be larger than the degree of v_i, therefore, if the value of *max* (size of the largest clique found so far) is not less than degree of v_i, then v_i can never lead to maximum clique of the graph. Similarly, a clique containing vertex v_i shall have vertices adjacent to v_i. Hence only the adjacent vertices of v_i are considered to obtain the largest clique containing v_i.

```
Algorithm : Find_Clique(U, size, C);
 1  begin
 2      while $U \neq \phi$ do
 3          if $size + |U| \leq max$ then /*Pruning 4*/
 4              return;
 5          end if
 6          select highest degree node u from U;
 7          $U \leftarrow U \backslash \{u\}$;
 8          $C \leftarrow C \cup \{u\}$;
 9          Find_Clique($U \cap Adj(u)$,size+1,C);
10      end while
11  end
```

Algorithm 5.1(b) Find_Clique ()

5.3.1 MCF Explained: An Example

MCF algorithm and its subroutine shown in algorithm 5.1 (a) and 5.1 (b) respectively are demonstrated with the help of an example.

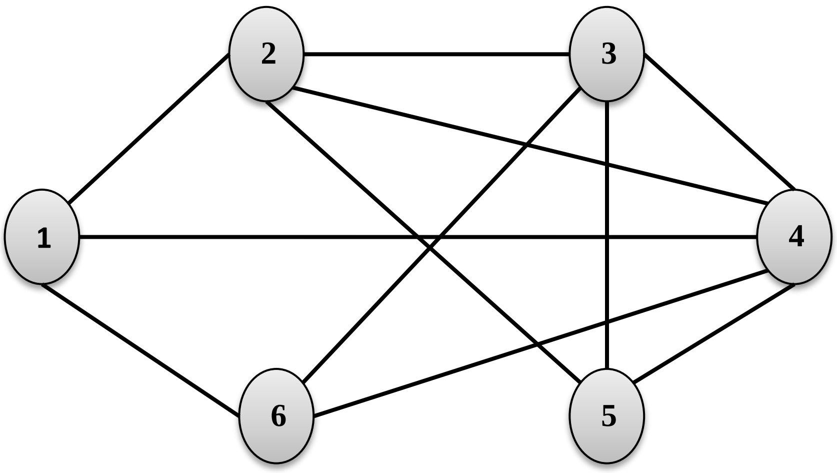


Figure 5.2 Example Graph

Consider a graph with 5 nodes and multiple edges as shown in figure 5.2. The working of MCF is illustrated in figure 5.3.

```
V = {4, 2, 3, 0, 1, 5} /*Sorted in non-decreasing order of their degrees*/
max = 0
size = 1
C = {4} /*highest degree vertex 4 is selected*/
Find_Clique({2, 3, 0, 1, 5}, 1, {4})
  C = {4, 2} /*vertex 2 is selected*/
  size = 2
  U = {3, 0, 1, 5}∩{1, 3, 4, 5}
  Find_Clique({3, 1, 5}, 2, {4, 2})
        C = {4, 2, 3} /*vertex 3 is selected*/
        size = 3
        U = {1, 5}∩{2, 0, 5, 4}
        Find_Clique({{5}, 3, {4, 2, 3})
          C = {4, 2, 3, 5} /*vertex 5 is selected*/
          size=4
          U = ∅∩{2, 3, 4}
          Find_Clique(∅, 4, {4, 2, 3, 5})
                return./*Main Function MCF*/
max = 4
M = {4, 2, 3, 5}
max = deg(2) /*Pruning 3 prunes rest of the vertices*/
return {4, 2, 3, 5} /*Max Clique*/
```

Figure 5.3 Illustration of MCF on Example Graph

This demonstration establishes the fact that MCF is an exact algorithm for finding maximum clique. It finds maximum clique with full correctness. The demonstration also provides justification to the pruning steps used in the algorithm. Due to pruning 3, MCF finds maximum clique in single iteration of the main algorithm which is extremely time effective. All 4 prunings reduce total computation in finding maximum clique significantly. MCF gives most time effective performance in finding maximum clique on various graphs as compared to other exact algorithms existing in literature [149, 150, 153]. The experiments and results comparison is discussed in chapter 6.

The above section has presented a time efficient exact algorithm to find maximum clique in a given graph. Maximum clique detection in a graph is an NP-Hard problem. This implies that the exact algorithm to find maximum clique in a large graph may take intractable time. There are two ways to handle the problem of intractability – one is to have a heuristic algorithm based on approximation, which is supposed to find a clique of size almost equal to the size of maximum clique in the graph. The second way is to have an exact algorithm with prunings. In later approach it is always guaranteed that the maximum clique is detected and pruning techniques used at various steps handle intractability. The algorithm proposed in this section i.e. MCF is based on the second approach. It significantly reduces time to find maximum clique in the graph using four pruning steps. It is an exact algorithm and finds maximum clique in every graph. However; the algorithm lacks in community detection which is broader domain of clique detection. Thus, the upcoming section proposes a novel approach for community detection i.e. K-Clique Based Community Detection using Union Find (KCUF).

KCUF is a community detection algorithm based on k-cliques. It exploits MCF for finding cliques. It also uses Union Find data structure for maintaining temporary communities. MCF and Union Find make KCUF significantly time efficient.

5.4 K-CLIQUE BASED COMMUNITY DETECTION USING UNION FIND (KCUF)

Identifying communities in social graph is one of the possible social network analysis strategies and it is most preferred. A social network may contain several communities. Community detection is the strategy to identify and locate the denser area in the social graph and determine the nodes in every denser area. These identified dense areas in the social graph are considered as communities. There is a possibility that there are two or more adjacent communities which may be close enough to be merged into one community. The community detection algorithm should take care of such cases as well. In dynamic social networks the communities change over the period of time. In case of dynamic social networks, the social network is continuously changing with time and therefore the corresponding social graph in the dataset is also changed accordingly. The associations continue to change in social graph which results in change in the communities.

Let’s assume that the social graph is maintained in snapshots. One snapshot of the social graph represents social network at one particular time. When associations change in social network, the changes are reflected in social graph and it is called new snapshot of the social network. Therefore, there is a possibility that the communities found in one snapshot of social graph may not be same as the communities found in the next snapshot. There are several events, which can result in formation of new communities or disappearance of existing communities in the social graph. Figure 5.4 represents various events [129] that may happen from the formation to disappearance of a community.

FORM: *If a new community is detected in snapshot k + 1 which was not present in snapshot k. This event is called FORM event where a new community is formed due to temporal changes in social network. In figure 5.4, community C4 is formed in snapshot k+1 which was not present in snapshot k.*

DISSOLVE: *A community is considered dissolved if it was present in snapshot k but not found in snapshot k + 1. Such situation is considered as DISSOLVE. Due to change in associations in the social network any existing community may not continue in future. In figure 5.4, community C5 gets dissolved.*

SURVIVE: *A community in snapshot k, if exists in snapshot k + 1 then it is considered to have survived the temporal changes in the social network. There is a possibility that few communities are common in snapshot k and k + 1. This event is called SURVIVE. In figure 5.4, communities C1 and C6 survive from snapshot k to snapshot k+1.*

SPLIT: *A community C in any snapshot may get split into two or more communities {C1, C2,….}. This means that if there are a atleast a threshold percentage of nodes from the a community C are majority of the nodes in communities {C1, C2, ...}, then the community C is considered to be split into {C1, C2,…}. In Figure 5.4, community C4 is split into two communities C5.1 and C5.2.*

MERGE: *In contrast to SPLIT event there is a possibility that two or more communities which existed in snapshot k may get merged into one community in snapshot k + 1. Most of the nodes from communities become part of one community. This event is called MERGE. In figure 5.4, communities C2 and C3 get merged into one community C23.*

Figure 5.4 demonstrates the change in snapshot due to happening of various events.

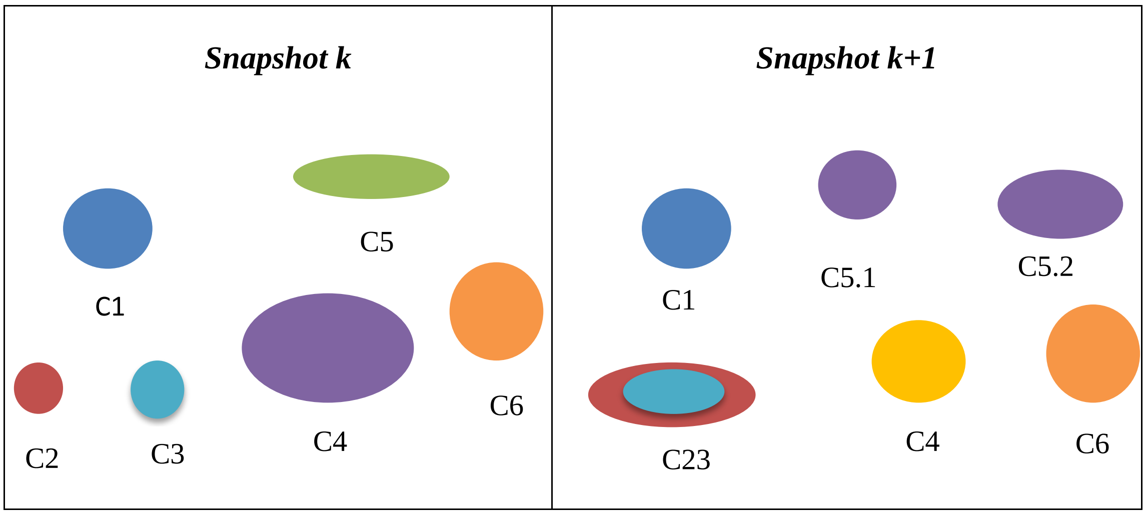


Figure 5.4 Events Related to Evolution of Communities

In case of dynamic social networks the community detection algorithm is responsible to detect the communities as well as to handle above mentioned events. The community detection algorithm can find a pattern of evolution of the communities by observing these events.

KCUF detects K-cliques in a social graph where K-clique is a clique of any graph with $k \geq 3$ vertices in it. The cliques which are adjacent to each other cannot be combined together to make a larger clique but they can be combined to form a bigger community. This is the basic idea on which KCUF works. Two cliques are said to be adjacent if they have $k - 1$ vertices in common. Similarly, two k-cliques are said to be reachable if they have chain of adjacent k-cliques between them. Reachability among k-cliques is achieved by transitivity of adjacency of k-cliques. If a k-clique K1 is adjacent to a k-clique K2 and K2 is adjacent to another k-clique K3 then K1 and K3 are reachable from each other. KCUF begins by determining all k-clique in the social graph and progresses by identifying reachable k-cliques and later combines them together into one community. This step is called grouping reachable k-cliques together to find bigger community in the graph. Value of *k* can be 3 or greater than 3 and it directly affects the size of the community and the number of the communities in the graph. KCUF determines adjacent k-cliques by intersecting them. Intersection operation is a time taking operation and it needs optimization for time efficient outcomes.

KCUF maintains following data structures which are exercised during execution of KCUF algorithm:

i. ***Node Map:*** *It is a mapping of node to cliques.* ***Node Map*** *of a particular vertex u is a set of all cliques having the vertex u. It is maintained for every vertex in the graph. This data structure is used to find overlapping cliques quickly.*

ii. ***Clique Map:*** *It is mapping of clique to cliques. Clique Map of a particular clique C is a set of cliques which are overlapping with C (atleast one node is common). Every clique is given a unique sequence number called* **clique_id**. *It is used to decide the order in which intersection between overlapping cliques (identified using Node Map) is performed.*

iii. ***Union Find:*** *It is a tree data structure which is very effective in finding membership of elements among sets to identify disjoint sets. Suppose there m sets having arbitrary elements. There is a possibility that few elements are commonly present in more than one set, thereby making them overlapping sets. There is another possibility that there are few sets which have no common elements to any of the other sets. Disjoint groups of sets can be formed on the basis of common elements and these are called disjoint sets. A disjoint set contains sets having same common elements Union Find data structure is time efficient ways to maintain and identify disjoint sets. Disjoint sets are maintained as union find trees. Each union find tree has a representative element. There are two operations possible in this data structure Union and Find. Union operation merges two union find trees if none of them is sub tree of the other one. Find operation is used to find representative element of the tree. If two trees have same representative elements, one is sub tree of the other then they need not be merged. Similar* *functionality is needed to decide whether two cliques are part of same community. Building Union Find data structure with n elements requires O(n) time and to find query is run in O(1) time.*

KCUF runs in two steps i.e. determination of cliques and determination of communities in the graph. The two step process is being illustrated as follows:

Step 1: Finding Maximal Cliques

i. *Execute MCF to find all maximal cliques.*

ii. *The objective of step 1 is to find all k-cliques in the graph so that they can be used to identify the communities in the graph. Communities are identified as group of reachable k-cliques in the graph. Reachability of cliques is determined by intersection operations. Step 2 explains the complete process of community detection using k-cliques in detail. Intersection operation is a time consuming operation. The time can be saved by reducing number of intersection operations done which directly depends on the number of cliques.*

iii. *MCF algorithm is used to find largest clique in the graph. The basic idea is to enumerate through every largest clique corresponding to each vertex. The largest among these is the maximum clique of the graph. Rest of the smaller cliques corresponding to other vertices is ignored.*

iv. *In KCUF, MCF does not ignore any clique. It enumerates through every largest clique corresponding to each node of the graph and all of these cliques are considered for community detection. MCF algorithm finds every maximal clique and puts into a set called Max Cliques.*

v. *The question is why maximal cliques and why not k-cliques? Because, any clique of size greater than k shall have two or more adjacent k-cliques. This means in maximal clique many reachable k-cliques are already combined and the time required for executing intersection operation to determine their reachability is saved significantly. An arbitrary graph of n nodes may contain up to $n^{n/3}$ maximal cliques [162].*

vi. *Henceforth, MCF finds all maximal cliques corresponding to every vertex in the graph.*

vii. *Every clique is given a unique sequence number.*

viii. *Node Map is maintained to map nodes to cliques to determine overlapping cliques.*

The result of this step is set of cliques called *Max Cliques*, of size atleast k, and *Node Map,* which maintains mapping of nodes to cliques. Next step iterates through all the cliques and determines the reachable cliques for merging into one community. Henceforth, Step 2 is called Community Detection.

Step 2: Community Detection

i. *Node map, prepared in step 1 is used to prepare Clique Map. Clique Map maintains clique to clique mapping, which means the cliques which have atleast one node in common mapped to one clique called representative clique of the map. Clique Map helps in finding which cliques should be intersected. Cliques which do not have anything common should not be intersected.*

ii. *Initially every maximal clique is considered as one independent community. Thus, total N communities are created (N is total number of maximal cliques discovered in step 1). Each represents a disjoint set, having a single maximal clique in it. The clique in the set is also its representative clique of this set.*

iii. *A list of communities is maintained called Community Array. All the maximal cliques are put into this array as disjoint sets. This array is converted into Union Find data structure which means, now Community Array contains disjoint sets which are iterated through to get similar sets merged together. The sets which are merged into any set are subsequently deleted from Community Array. At the end Community Array shall have all disjoint sets of cliques each representing an individual community of the graph.*

iv. *Each maximal clique is iterated one by one and it is intersected with the representative cliques of the Clique Maps. Initially every clique is representative of its own Clique Map. When a new clique C is to be* *determined for its community, Find Set operation is used to find representative clique R of the Clique Map of C (in first iteration C and R are same) and it is intersected with every other representative maximal clique in Community Array. If R and representative of any Clique Map CM in Community Array has k-1 nodes in common, these two Clique Maps are merged using Union operation.*

```
Algorithm : KCUF(G)
    G(V,E) : Input Graph
    Max_Cliques : Set of maximal cliques found in G(V,E)
    Community_Array : Map for Union_Find
    Node_Map(u) : Set of Cliques having node u
    Clique_Map(C) : Clique to clique map
    begin
        Max_Clique = MCF(G);
        for each maximal clique C ∈ Max_Clique
            for each node u ∈ C
                update(Node_Map , u , C);
            end for
        end for
        for each node u ∈ Node_Map
            for each maximal clique C ∈ Node_Map(u)
                update(Clique_Map, C, Node_Map(u));
            end for
        end for
        BuildUnionFind(Community_Array, Clique_Map);
        while(Max_Clique ! Empty)
            $C_i$ = Extract_Min(Max_Clique);
            for each maximal clique $C_j$ ∈ Clique_Map($C_i$)
                if(FindSet($C_i$) != FindSet($C_j$) && $n(C_i \cap C_j) == k - 1$)
                    Union(Community_Array($C_j$) , Community_Array($C_i$));
                end if
            end for
        end while
        return Community_Array;
    end
```

Algorithm 5.2 K-clique based community detection using Union Find (KCUF)

v. *Community Array is updated accordingly with the size of the community. This process is repeated until all maximal cliques are traversed. Community Array contains every community discovered by KCUF.*

Algorithm 5.2 shows the working of KCUF.

MCF and KCUF algorithms work in collaboration to handle large social graphs for community detection. In chapter 6, the efficiency and correctness of these algorithms is investigated on social network data collected and on some standard graph data sets. The results are compared with already established techniques in the same fields.

5.5 SUMMARY

In this chapter, a time efficient maximum clique finding algorithm, MCF, and a community detection framework based on k-cliques, KCUF have been proposed. MCF is an exact algorithm to find maximum clique in given graph which utilizes pruning to eliminate the unrewarding branches of computation leading to time-efficient MCF.

KCUF is a k-clique based community detection algorithm. KCUF identifies communities by merging adjacent k-cliques. KCUF reduces time required to perform union operations significantly. KCUF deploys MCF to identify all maximal cliques in the graph in first step. The maximal cliques identified by MCF are considered as initial communities. Adjacent communities are then merged using union operation. Union Find data structure is used to maintain list of communities. KCUF successfully detects communities in a given graph in a time efficient manner.

Next Chapter is dedicated to discussion about implementation of proposed work and the results thus obtained.

Chapter 6

RESULTS AND DISCUSSION

The chapter presents the details pertaining to the implementation and results of work proposed in chapter 4 and chapter 6. The work began by proposing data collection technique and later was extended by incorporating the crawler, CCSC. On identifying the limitation of CCSC, the same was further extended to FPBC. While the implementation of FPBC was in progress, it was realized that community detection is also a prevailing challenge in social network analysis. Hence, KCUF integrated with MCF was proposed and implemented. In addition to this package a parallel module TINB has been developed to realize non-social domains as traditional social networks. In this chapter, the implementations of all proposed techniques, their applicability, correctness and efficiency are being presented and are also investigated against already established techniques in the allied domain in literature. The techniques and frameworks have been tested on existing social networks as well as on synthetic social networks (exclusively in few cases). The standard graph datasets [122, 151, 163 - 168] have also been used to establish the correctness and efficiency of proposed algorithms. Upcoming section discusses the implementation of CCSC, FPBC, TINB, MCF and KCUF.

6.1 IMPLEMENTATION DETAILS

6.1.1 Cluster Coverage Sampling Crawler (CCSC)

CCSC has been implemented using Java and HTML Unit API. CCSC is responsible for collecting sample from large social network for analysis. It is already mentioned that CCS algorithm has been employed as sampling algorithm in CCSC. CCS does not suffer from biasing and also reflects original clustered view of the social network in the sample collected. To establish this fact CCSC has been tested on a synthetic social network created on local server. CCSC has been first tested on a synthetic network so that the claimed elimination of biasedness and coverage of clustered view can be verified. It is worth mentioning that the listed properties cannot be verified on the original online social network because of its enormous size as until the complete social network is known, the same cannot be verified. Therefore, a synthetic social network has been prepared which has similar characteristics such as power law of degree distribution, clustering coefficient, just to list a few, as does by any traditional social network like Facebook or Twitter. The actors of a social network are given full profiles of attributes such as place of birth, place of work, education, political orientation, birthday etc. These attribute have their inherent values. The attributes of users and their values are used for creating Boolean expression EXP. The statistics of the synthetic social network is shown in table 6.1

Table 6.1 Statistics of Synthetic Social Network

Entity	Count
Number of Nodes	1539
Number of Edges	20234

The attribute values of each node in the network are created anonymously. CCSC is run on this synthetic social network. The crawler is run and paused after some time to collect a smaller sample. Then the crawling is resumed to collect a larger sample. Table 6.2 shows statistics of both samplers.

Table 6.2 Samples statistics

Sample	Number of Nodes	Number of Edges
Sample 1	257	1332
Sample 2	513	4442

Two samples are collected to establish the fact that CCSC does not lose the direction of crawling and both samples demonstrate sufficient resemblance with the original network. To quantify the similarity between original social network and the collected samples, degree distribution [75] is taken as criterion where degree distribution in the social network is direct measurement of structural characteristics of any social network as it directly affects modularity in the social graph and community formation in the network. Degrees of every node in original social network and in samples are normalized using equation 6.1. Normalization is required to have degree distribution within a closed range. In this work median of degrees of all nodes is being used for normalization and after normalization it falls in range of 0 to 2.

$$\text{Normalized\_Degree}_j = \frac{\text{Degree}_j}{M} \tag{6.1}$$

Where, M is median of degrees in the network and $1 \leq j \leq$ total number of nodes.

Degree distribution of original social network and the samples are shown in figure 6.1, where x-axis depicts normalized degree and y-axis denotes fraction of nodes.

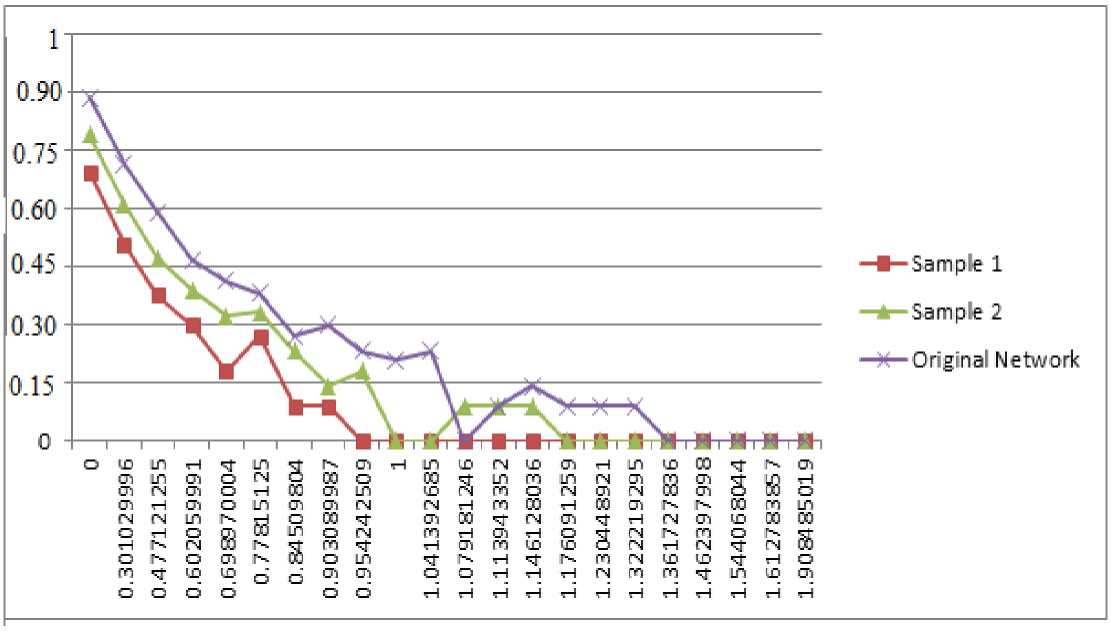


Figure 6.1 Degree Distribution of Samples and Original Network

It is evident in the graph itself that the degree distribution in both samples is similar to the degree distribution in original network.

To compare the correctness and efficiency of CCS algorithm used by CCSC, the synthetic network is being sampled using BFS [63, 64], NBRW-rw [70] and MHRW [63, 70, 71] sampling algorithms. These algorithms are popular in the field of sampling of social networks. The similarity between degree distribution of sample and degree distribution in original social network is being used as metric for correctness. To quantify the error (dissimilarity) in degree distributions, Normalized Mean Squared Error (NMSE) [75] is being used. Equation 6.2 represents computation of NMSE of a node degree k.

$$\text{NMSE}(k) = \frac{\sqrt{E\left[\left(\hat{\theta}_k - \theta_k\right)^2\right]}}{\theta_k} \tag{6.2}$$

Where, $\hat{\theta}_k$ is the estimation of θ_k based on the sampled graph. NMSE(k) is defined in order to show the difference between the degree distribution of the sampled graphs and the original social graph. The lower the NMSE value of an algorithm, better is performance of the sampling algorithm in the social network graph.

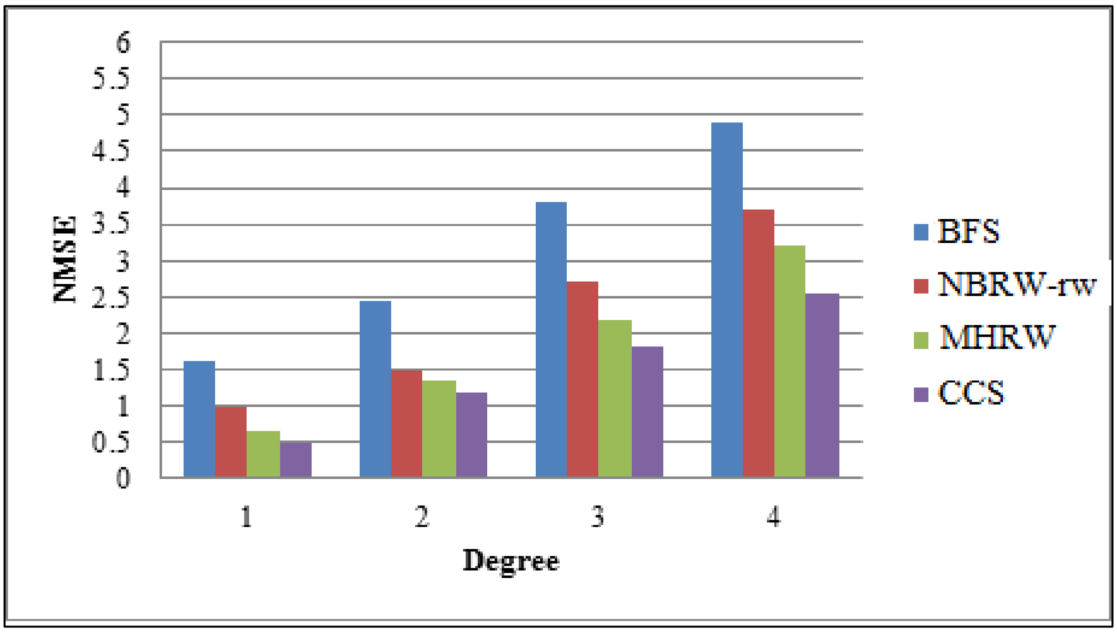


Figure 6.2 NMSE Comparison Graph

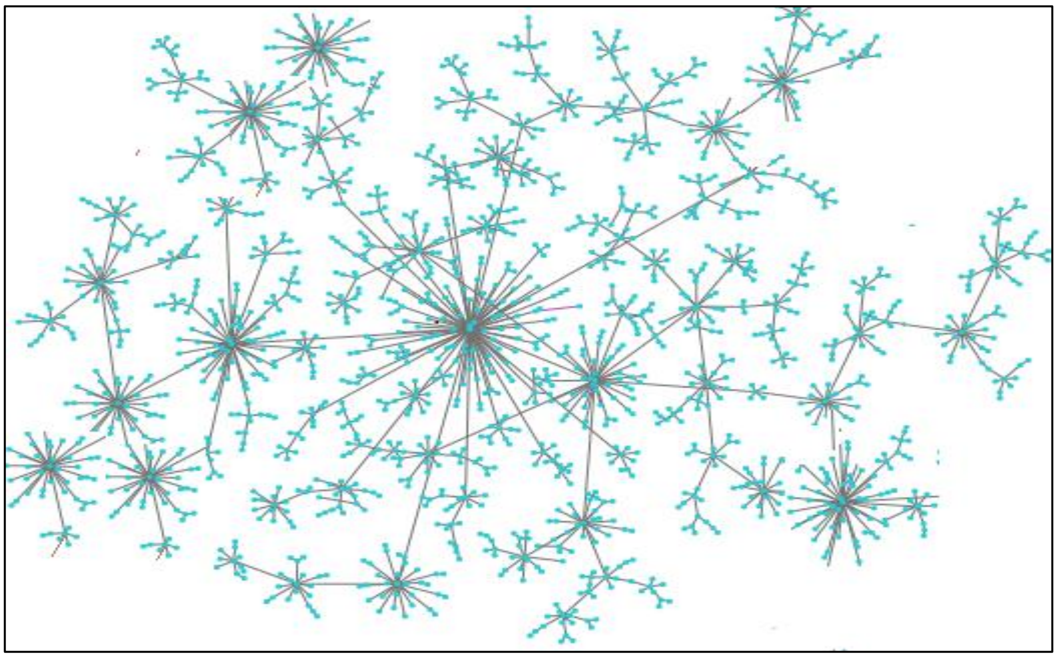
It is clearly evident in the graph shown in figure 6.2 that the performance of CCS algorithm is better as compared to other significant methods of sampling the social network.

Figure 6.3 (a) – (c) shows visualization of original social network sample 1 and sample 2 respectively. It is clearly evident the original network is clustered and is having overlapping clusters too. Similar overlapping clustered view can be seen in sample 1 and sample 2. All the visualizations are done using Net Logo Tool[3].

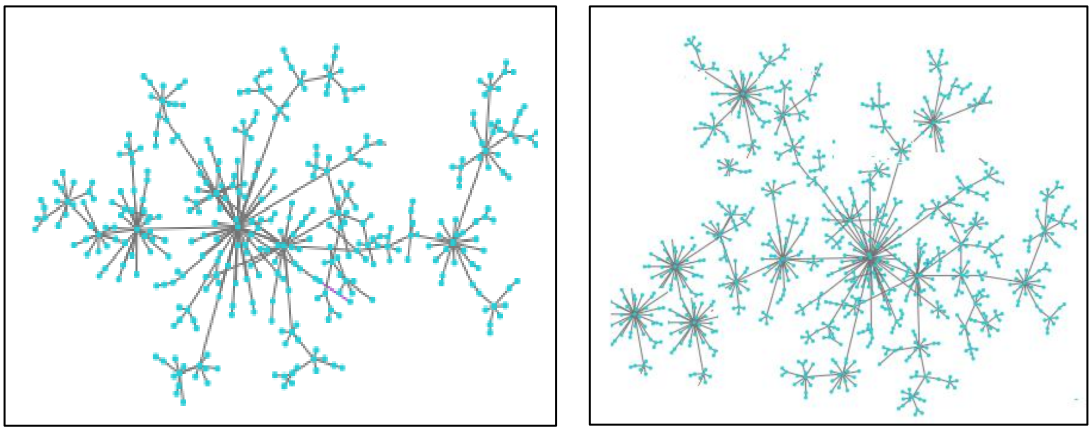
Above discussion establishes the fact that the proposed social network crawling framework CCSC exhibits superior performance as compared to other famous sampling techniques. CCSC is able to capture the clustered structure of the social network without getting biased towards similar nodes. Two samples of the same social network are captured and both are aligned with the structural properties of the

[3] https://ccl.northwestern.edu/netlogo/

original social network. It is obvious that the size of the sample directly depends on the time for which CCSC is run. If the social network is finite then the representation extent of the clusters will increase with time after coverage of every cluster.



(a)



(b) (c)

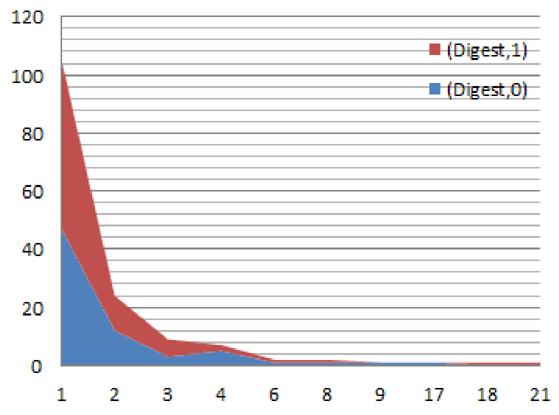
Figure 6.3 (a) Clustered View of Original Social Network (b) Clustered View of Sample-1 (c) Clustered View of Sample-2

But in practicality, the size of real time social networks is gigantic and thus more is the time given to CCSC, more would be formation of clusters. In this regard, the frequency of changing the expression also affects the size of clusters in the sample. If the expression is changed frequently then the crawler gets less time to stay is a cluster.

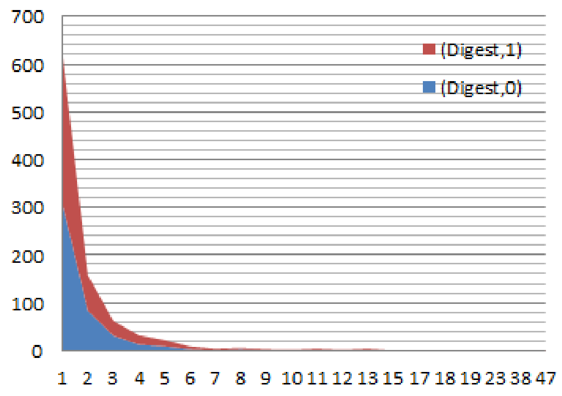
The crawler is terminated on gathering sufficient number of nodes and their relationship edges. The current work considers *10%* of friends of sampled node as the termination condition The collected sample shall cluster alike original network, thereby giving maximum possible representation of the network.

6.1.2 Foot Prints Based Crawler (FPBC)

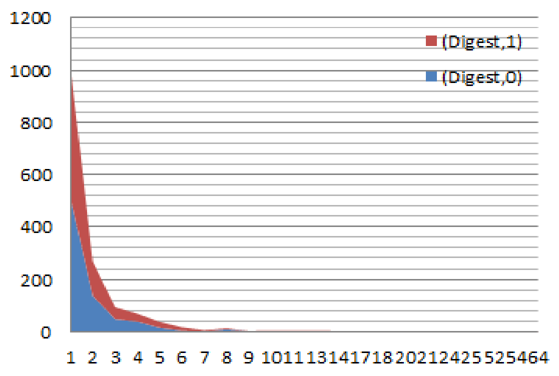
FPBC is an extension of CCSC. It is being implemented in Java using HTML Unit API. Two iterations of crawler are being performed. First iteration is considered as base iteration. In base iteration, sample of the social network is collected and log is created, which is called *Foot_Print*. The sample contains nodes and their association information. Every Foot Print contains timestamp i.e. logical time for log creation, Boolean expression for sampling nodes and the digest containing compressed structural information of sample i.e. degree distribution in the sample collected so far. FPBC is run on Facebook. Almost 5000 nodes are crawled in the base iteration of the crawler.



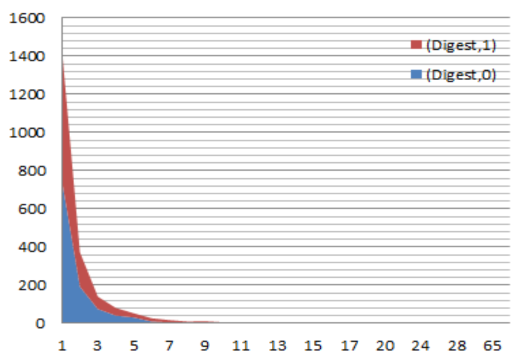
(a) Degree Distribution of Digests Logged at t1



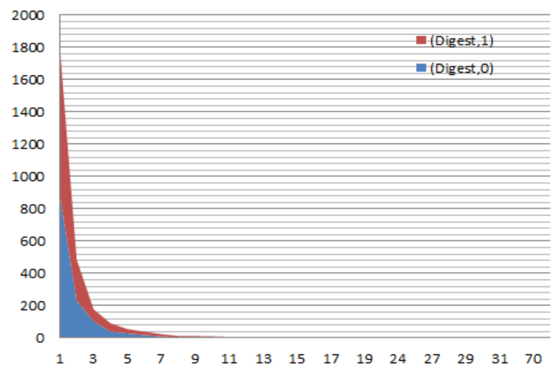
(b) Degree Distribution of Digests Logged at t2



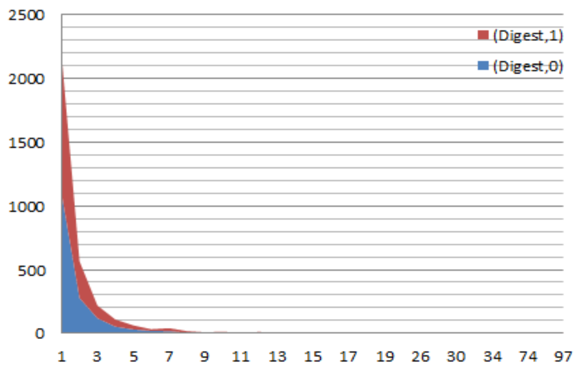
(c) Degree Distribution of Digests Logged at t3



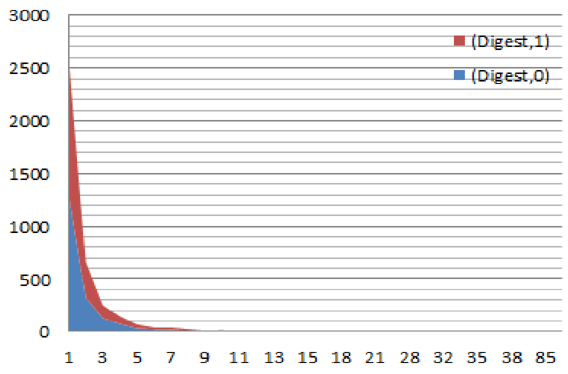
(d) Degree Distribution of Digests Logged at t4



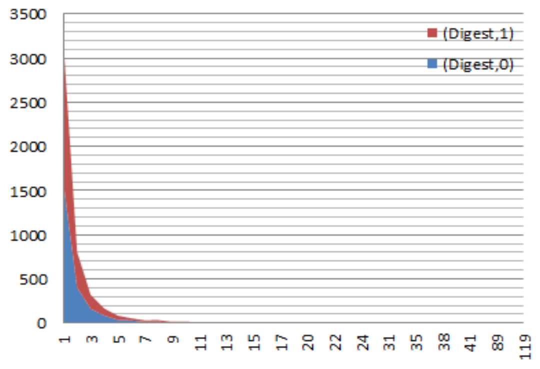
(e) Degree Distribution of Digests Logged at t5



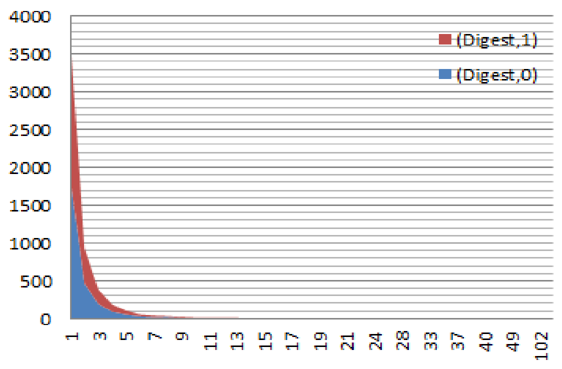
(f) Degree Distribution of Digests Logged at t6



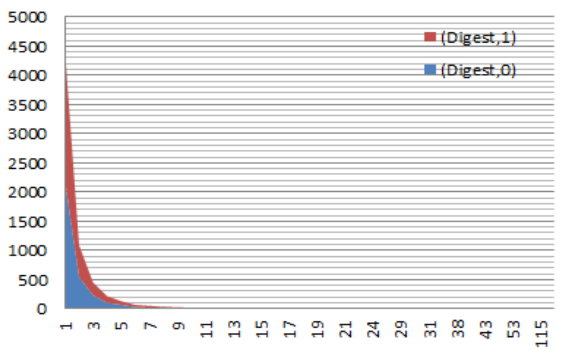
(g) Degree Distribution of Digests Logged at t7



(h) Degree Distribution of Digests Logged at t8



(i) Degree Distribution of Digests Logged at t9



(j) Degree Distribution of Digests Logged at t10

Figure 6.4 Degree Distribution of Digests Logged at Regular Intervals During Crawling

Ten Foot Prints are logged in Foot_Print during base iteration. FPBC is run again with the same seed node second time. In this iteration also around 5000 nodes are crawled. This iteration is different from base iteration because in base iteration FBPC runs autonomously but in in next iteration, the crawler is governed by Foot_Print. During second iteration the freshness of the sample is verified. Figure 6.4 (a) – (j) show comparison of degree distribution of digests collected during base and second iteration respectively at various times where, x-axis represents degree and y-axis represents number of nodes (in Thousands).

The accuracy of any social network analysis majorly depends upon the degree distribution in original network and in the sample as well. If the degree distribution is similar then the sample is considered to be a good representative of the original network.

This justification has been used to measure similarity between digests. If the digests created at certain time stamp exhibit similar degree distribution then the data set is fresh. To calculate dissimilarity between digests, NMSE in degree distribution of both digests has been used as performance metric. NMSE has been calculated using equation 6.3.

$$\mathrm{NMSE}(t_i) = \frac{1}{N}\sum_{k=1}^{N}\frac{(P_k - M_k)^2}{\bar{P}\bar{M}} \qquad (6.3)$$

Consider digests at time t_i, $(Digest, 0)_{t_i}$ is base digest and $(Digest, 1)_{t_i}$ is latest collected digest. P_k is number of nodes having degree k in $(Digest, 0)_{t_i}$ and M_k is number of nodes having degree k in $(Digest, 1)_{t_i}$. $\bar{P}$ and $\bar{M}$ are average degrees of each node in $(Digest, 0)_{t_i}$ and $(Digest, 1)_{t_i}$ respectively.

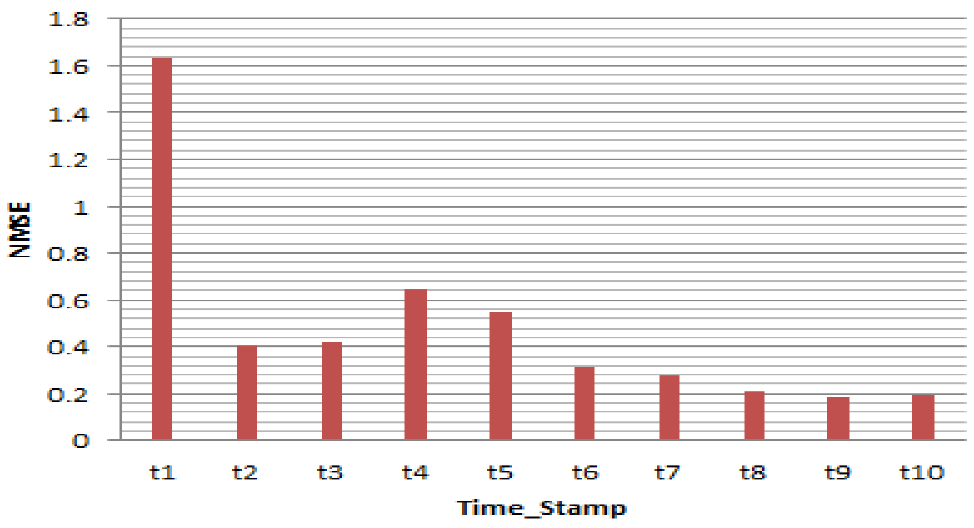
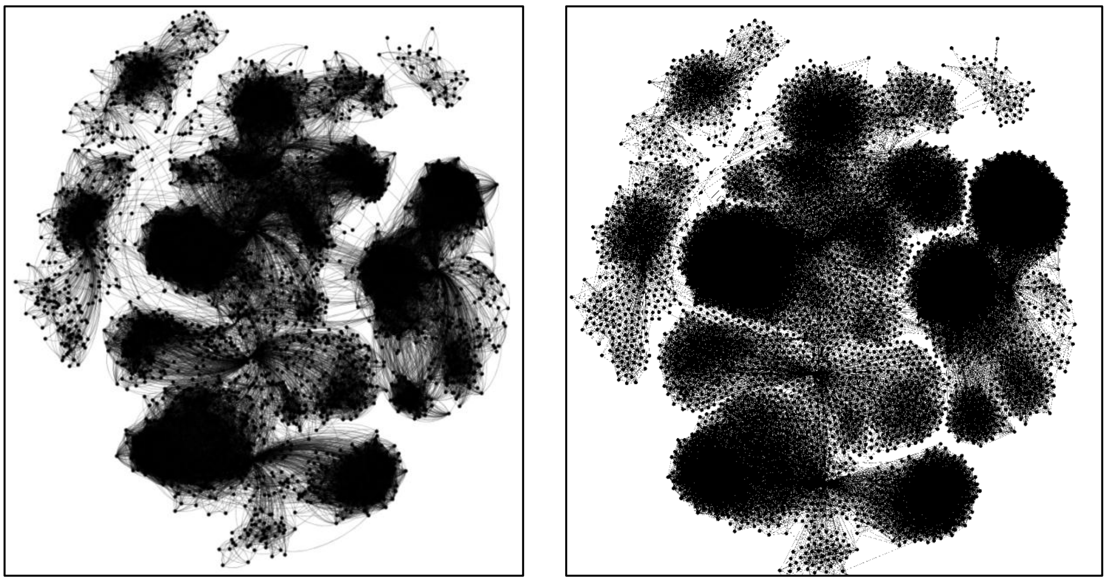


Figure 6.5 NMSE Calculated for Degree Distribution in Base Sample and Sample Collected in Next Iteration.

Figure 6.5 represents NMSE of two digests calculated at different time stamps. It is evident that NMSE between various time stamps is different and random. Overall dissimilarity is measured by average NMSE calculated using equation 6.4.

$$\text{Average NMSE} = \frac{1}{n}\sum_{i}^{n}\text{NMSE}(t_i) \tag{6.4}$$

Average NMSE is cumulative measure of freshness of the sample dataset. As per the analysis, the threshold value *h* of average NMSE can be declared and the dataset is updated only if average NMSE crosses *h*. Different social networks may have different rate of dynamism (γ) where γ implies the rate at which the sample in the dataset gets obsolete. In other words, γ of a social network represents pattern of changes in the network. Once the pattern is identified, time for next crawl can be decided. The pattern of evolution of a social network is quantified by two parameters viz. Average NMSE between latest crawled dataset and previously crawled dataset and duration between the crawls [42]. Suppose the duration between two crawls is *d* and average NMSE between datasets crawled in consecutive iterations is *h'*. If *h' > h* then duration for the next crawl is reduced to *d/2*.This practice is repeated until we get *h' <= h*. Now we know the duration of next crawl should be set as twice of duration of the last crawl. Figure 6.6 (a) and (b) shows initial sample dataset and dataset after second iteration.



(a) (b)

Figure 6.6 (a) Sample Dataset in first iteration of the crawler. (b) Refreshed sample dataset after second iteration of the crawler.

Two iterations of the crawler are run and freshness of the sample dataset is put on test under the mechanism proposed in this research work. Two sample datasets are visualised using SNA tool Gephi[4].The visualization of both datasets establishes the fact that the crawler in second iteration maintains its path overlapping the footprints it left in first iteration. The proposed framework thus achieves the intended aim.

6.1.3 Maximum Clique Finder (MCF)

Maximum Clique Finder (MCF) algorithm is dedicated to SNA for clique detection. MCF is a time efficient maximum clique finding algorithm primarily based on pruning. The current section presents the results obtained on implementing MCF and it also compare the performance of MCF with other similar algorithms. MCF is being implemented on 64 bit windows 7 Home Basic with 2.3 GHz Intel Core i3 with 32 GB of main memory with NeuTroN DoS-C++ version 0.77.0.0. Simple adjacency list representation for graph is being maintained as a reference array of size |V|, which contains references for |V| lists of vertices, each corresponding to a particular vertex. Figure 6.7 shows the representation of adjacency list.

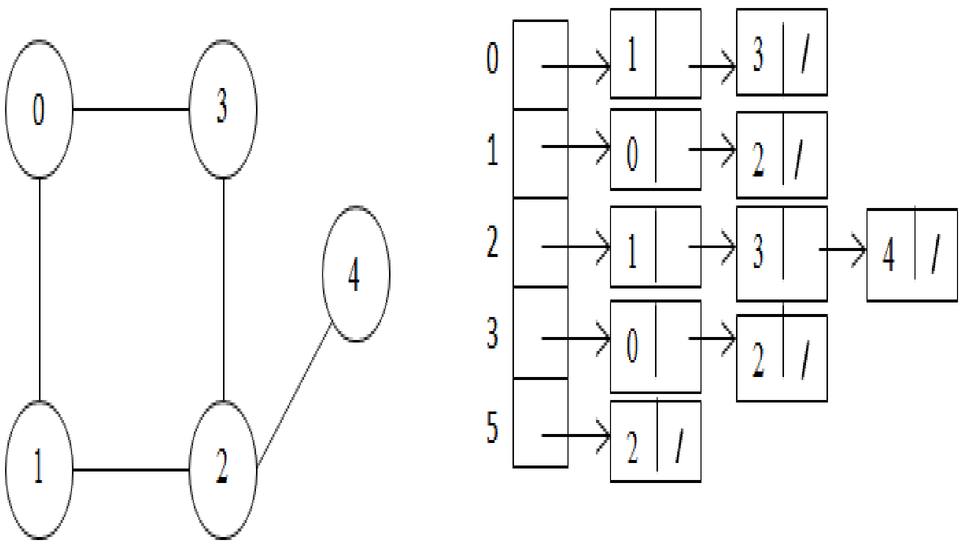


Figure 6.7 Adjacency List

Two categories of graphs are chosen for experiments. First includes graphs originated from real world applications. Table 6.3 gives brief description of such graphs. Second category includes graphs from DIMACS Implementation Challenge [151].

[4] https://gephi.org/

Table 6.3 Description of Real World Graphs

Graph	Description
Email_enron [122]	Communication network of E-mail Exchange
dictionary28 [163]	Network of words
code-mat-2003 [164]	Collaboration network of scientists
foldoc [165]	Dictionary for computing related terms
web-Google [166]	Web graph released as part of Google programming contest
soc-wiki-vote [167]	Wikipedia page vote network
daysall [168]	Network of Reuter terror news obtained from CRA networks

Table 6.4 Structural Characteristics of Graphs

Graph	|V|	|E|	Edge Density	Max Degree
Email_enron	36692	183831	0.00027	1482
dictionary28	52652	89038	0.00006	38
code-mat-2003	31163	120029	0.00025	202
foldoc	13356	91470	0.00103	728
web-Google	916428	4322051	0.00001	6332
soc-wiki-vote	8297	100762	0.00293	1065
daysall	13308	148035	0.00167	2265
hamingo6-4	64	704	0.34921	22
c2000.5	2000	999836	0.50010	1074
c4000.5	4000	4000268	0.50000	2123
Johnson8-94	70	1855	0.76812	53
keller4	171	9435	0.64912	124
le450_25	450	17343	0.17100	179
le450_25d	450	17425	0.17200	157
c-fat200-5	200	8473	0.42578	86
brock200-2	200	9876	0.49628	114
r250-5	250	14849	0.47700	191
r1000-1	1000	485090	0.97100	991

Table 6.5 Comparison of Runtimes of Various Algorithms. Δ is Size of Max-Clique Found. $\boldsymbol{T_{CP}}$, $\boldsymbol{T_{Ostergard}}$, $\boldsymbol{T_{MCQD}}$ is Time Taken by Carraghan Pardalos, Ostergard and MCQD Algorithms Respectively. $\boldsymbol{T_{MCF}}$ is Time Taken by MCF Algorithm.

G	Δ	T_{CP}	$T_{Ostergard}$	T_{MCQD}	T_{MCF}
Email_enron	20	6.940	16.210	4.010	1.001
dictionary28	26	7.154	28.254	8.102	0.100
code-mat-2003	25	8.010	12.003	1.960	0.021
foldoc	9	*	3.210	0.620	0.070
web-Google	44	*	11006.070	*	1.203
soc-wiki-vote	17	0.720	0.940	0.310	2.014
daysall	28	8.440	10.010	1.002	0.210
hamingo6-4	4	<0.01	<0.01	<0.01	<0.01
c2000.5	184	40.810	26.354	8.207	32.112
c4000.5	247	378.24	101.245	24.200	87.024
Johnson8-94	14	0.201	<0.01	0.011	0.310
keller4	11	26.610	0.220	0.031	0.010
le450_25	25	0.120	0.100	0.100	<0.01
le450_25d	25	0.120	0.100	0.100	<0.01
c-fat200-5	58	0.710	0.345	0.010	0.556
brock200-2	12	1.022	0.032	0.010	0.540
r250-5	35	0.010	0.010	0.010	0.010
r1000-1	291	2.214	0.997	0.445	0.210

Table 6.4 represents structural properties of the graphs from both categories. Performance of MCF is compared with three popular algorithms i.e. Carraghan

Pardalos [149], Ostergard Algorithm [150] and MCQD [153]. All the listed algorithms have been implemented on similar platform as mentioned above for MCF and the results are being reproduced in table 6.5. The current work sets an upper limit of 1800 seconds on runtime. The program is forcefully aborted if it fails to terminate in 1800 seconds. It is shown by an Asterisk (*) in the table 6.5. The time is in seconds.

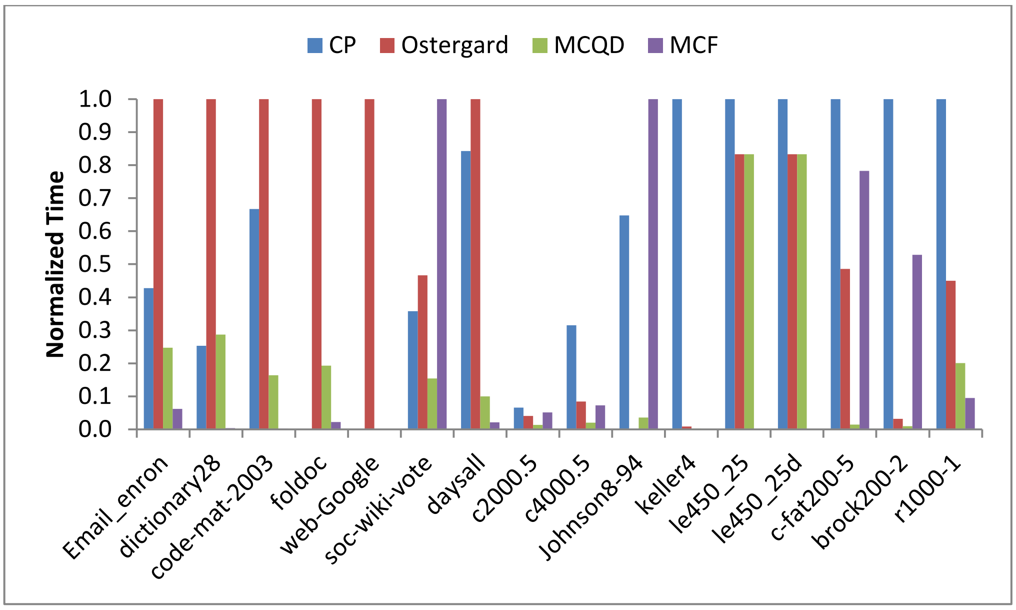


Figure 6.8 Runtime (normalized to slowest algorithm) comparison of various algorithms

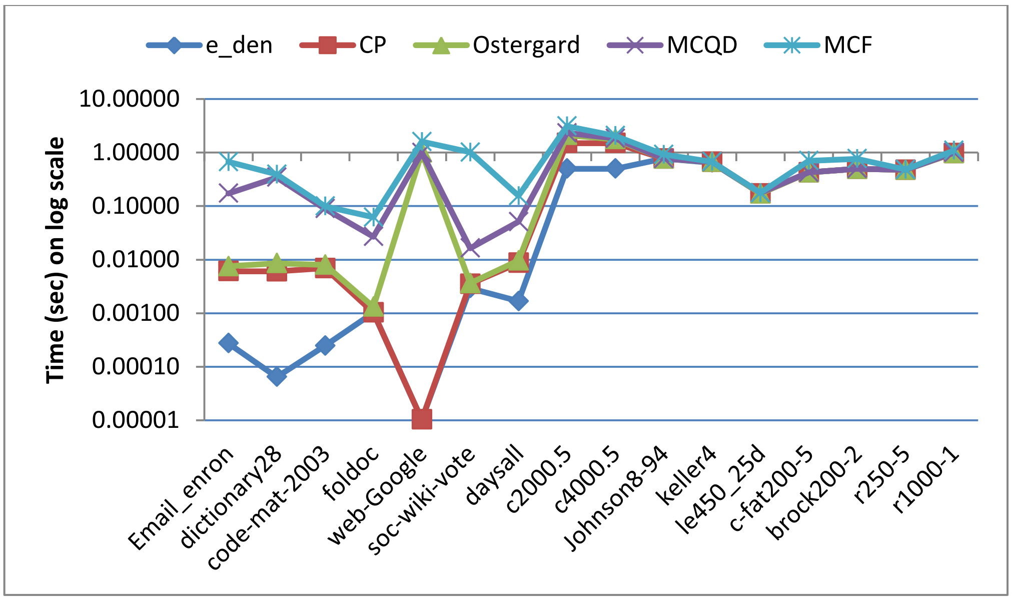


Figure 6.9 Runtimes of various algorithms vs edge density.

A comparative representation of times is shown in figure 6.8 where the time is normalized to fall in the scale of 0 to 1. Time taken by various algorithms is also compared with respect to the edge density of the graph. Figure 6.9 represents comparison of runtimes of various algorithms against edge density in the graphs in which it is clearly evident that MCF performs excellent if the graph is sparse.

The above discussion of the results establishes the fact that in most cases MCF has performed better than other three algorithms. It concludes that MCF is the most time efficient algorithm for finding maximum clique in a graph. Edge density of the graph directly affects performance of clique finding algorithm. All four algorithms have been tested with respect to the edge density of the graph and it is found that MCF performs better if the edge density is low in a graph. Almost every social graph follows power law of degree distribution [9, 12], which implies that there are small number of higher degree vertices and large number of lower degree vertices. Pruning in MCF utilizes this characteristic and most of the lower degree vertices are pruned in initial iterations of the same. The algorithm produces very good results for DIMACS graphs as well.

A direct application of MCF is seen in community detection in social graphs. While both clique and communities represent denser areas but the two terms have different degree of density. Henceforth, it can be said that determining cliques in the graph can pave the way to find communities in the graph. In the next section, a novel algorithm namely KCUF for community detection is being implemented.

6.1.4 K-Clique based Community Detection using Union Find (KCUF)

KCUF is being tested on two significant real time datasets, Cond-MAT[5] and Amazon joint purchasing dataset[6]. These are baseline datasets and are used in various applications. The measurement of correctness of the framework depends on the number of communities detected in every run. The efficiency of KCUF is compared with two significant community detection algorithms KUM [132] and REID [136] respectively. Bothe KUM and REID algorithms are also based on k-cliques. The

[5] https://arxiv.org/archive/cond-mat

[6] https://snap.stanford.edu/data/com-Amazon.html

number of communities detected in the graph depends on the value of k. Therefore, the experiments are for different values of k. The experiments performed and results obtained on Cond-MAT dataset and Amazon dataset are being discussed in subsequent sections.

A) Community Detection using Cond-MAT

Cond-MAT dataset is obtained from e-print arXiv library which was built in August, 1991 in National Laboratory of American Los Alamos. Cond-MAT contains 1,25,959 edges and 30,561 nodes. KCUF, KUM and REID algorithms are run on this dataset for different values of k. The communities detected by each algorithm are shown in table 6.6. It is worth noticing that KUM, REID and KCUF as well could detect same number of communities, which proves correctness of KCUF.

Table 6.6 Number of Communities for Different k in Cond-Mat Dataset

k	**KUM**	**REID**	**KCUF**
3	3569	3569	3569
4	4409	4409	4409
5	3319	3319	3319
6	2449	2449	2449
7	1765	1765	1765
8	1131	1131	1131
9	672	672	672
10	392	392	392
11	178	178	178

The distribution of communities of different scales in Cond-MAT dataset are shown in figure 6.10, for k = 4. It is evident in the graph that the communities are uniformly distributed except few. The correctness of the KCUF is justified by above discussion. The efficiency and effectiveness of KCUF is determined by comparing time taken to detect the communities.

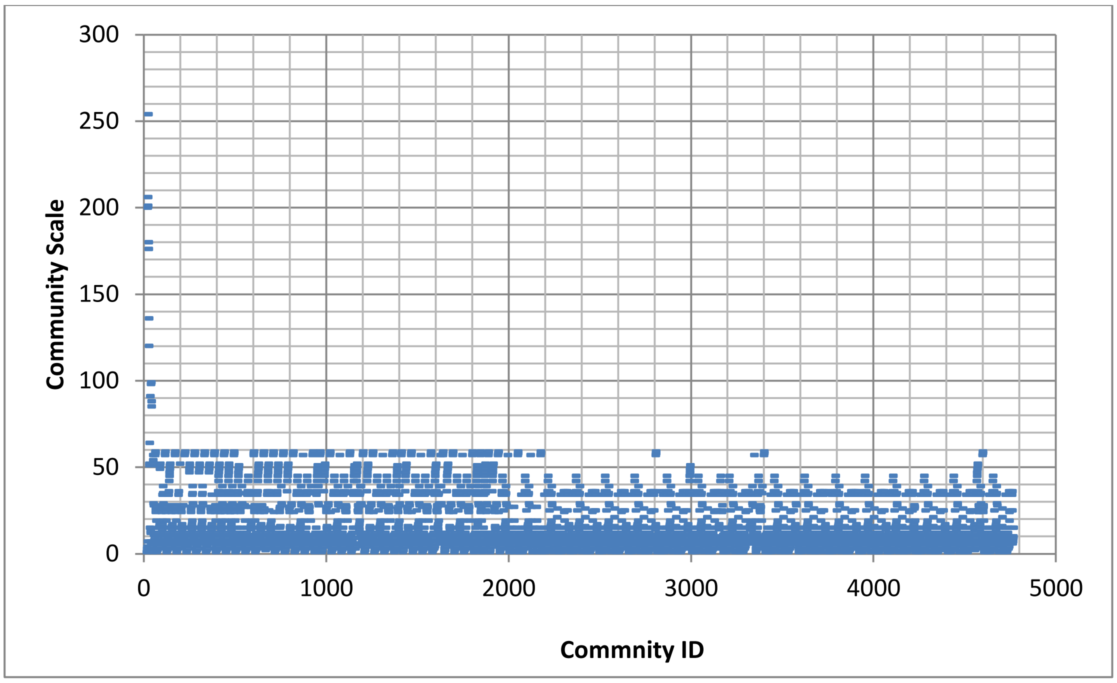


Figure 6.10 Community Distribution in COND-MAT for k=4

Figure 6.11 indicates time taken by KCUF, KUM and REID algorithms.

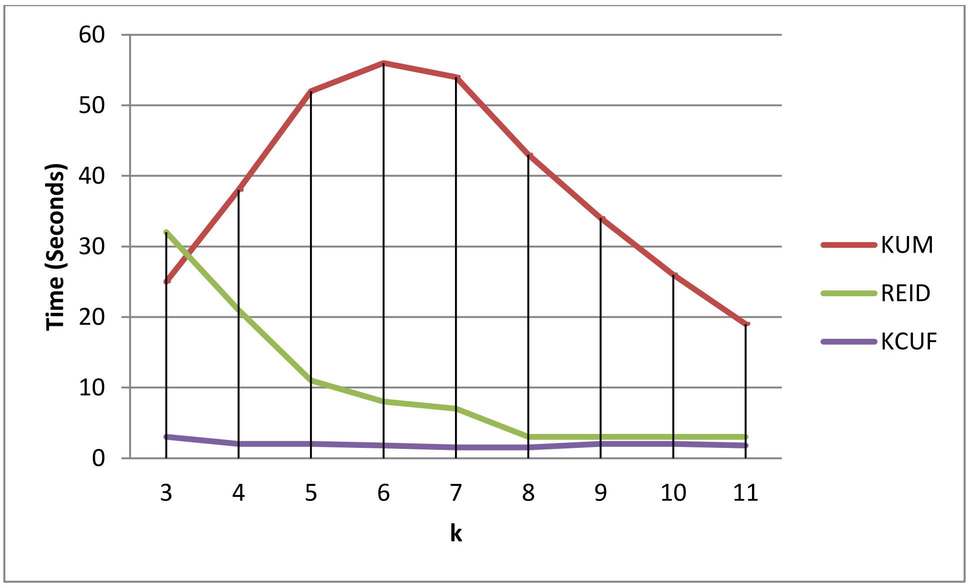


Figure 6.11 Time Taken to Detect Communities in COND-MAT for Different Values of k

It is observed that KCUF outperforms the other two algorithms for different values of k. KCUF takes less time to detect communities of any value of k as compared to time taken by other algorithms. KUM and REID algorithms show different pattern of time consumption for different values of k. While KUM and REID show monotonic pattern of time for different values of k, KCUF initially reflects upsurge in the time with the increasing value of k and then the time reduces. But in every case KCUF algorithm performs better than both of these algorithms for every value of k.

B) Community Detection using Amazon Product Online Joint Purchasing

Amazon online sales dataset is prepared on the basis of online purchase of goods. If two products are bought together then these two products are considered to be associated with each other. For instance, if a product p is purchased with another product q then p and q are associated and a connection edge between p and q is added in the graph of goods. This dataset is downloaded from online repository of social graphs SNAP. It has 9,25,872 edges and 3,34,863 nodes. Table 6.7 shows results of community detection by KUM, REID and KCUF algorithm for different values of k respectively.

Table 6.7 Number of Communities in Amazon Goods Graph for Different k

k	**KUM**	**REID**	**KCUF**
4	23234	23234	23234
5	10842	10842	10842
6	2631	2631	2631
7	31	31	31

Amazon goods network is larger and more complex than Cond-MAT network. It can be seen in observed from table 6.7 that all three algorithms are able to find same number of communities. This fact indicates that KCUF is accurate even for larger and more complex graphs. . Figure 6.12 reflects that the communities are evenly distributed except few k = 6.

Figure 6.13 shows time consumption of each algorithm. It is observed that KCUF performs better than other two algorithms for each value of k. For k = 4, the time consumption of REID algorithm is 1461.72 sec, while KUM algorithm takes 92.21 sec, but KCUF takes 19.1 sec. In addition, KUM is faster than REID till k = 6. After k = 5, REID beats KUM but KCUF is best for any value of k. Figure 6.13 shows time consumption of each algorithm. It is observed that KCUF performs better than other two algorithms for each value of k. For k = 4, the time consumption of REID algorithm is 1461.72 sec, while KUM algorithm takes 92.21 sec, but KCUF takes 19.1

sec. In addition, KUM is faster than REID till k = 6. After k = 5, REID beats KUM but KCUF is best for any value of k.

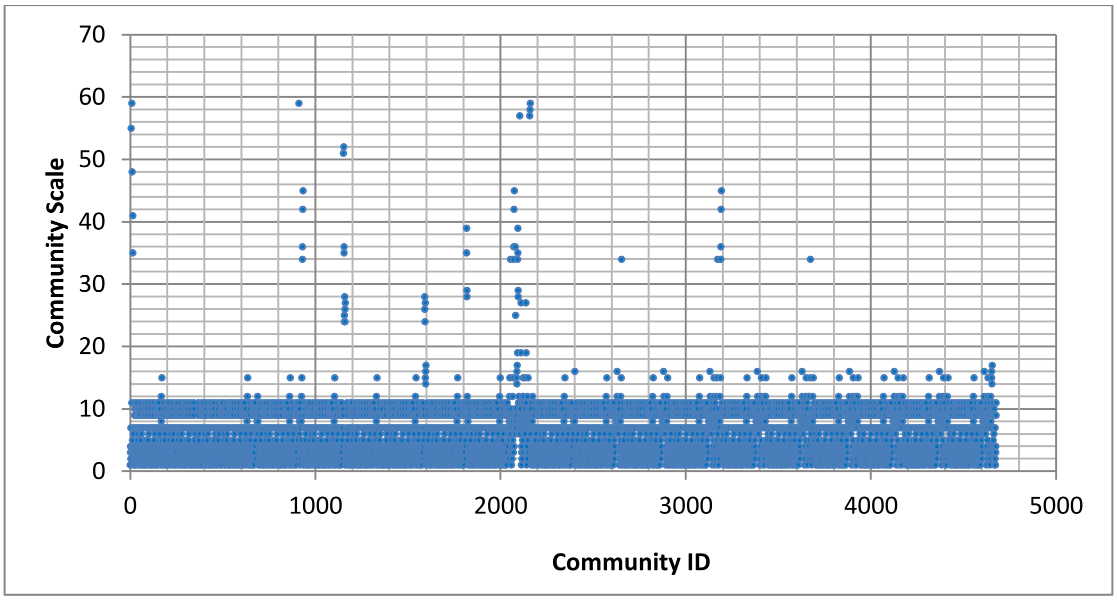


Figure 6.12 Community distribution in Amazon for k=6

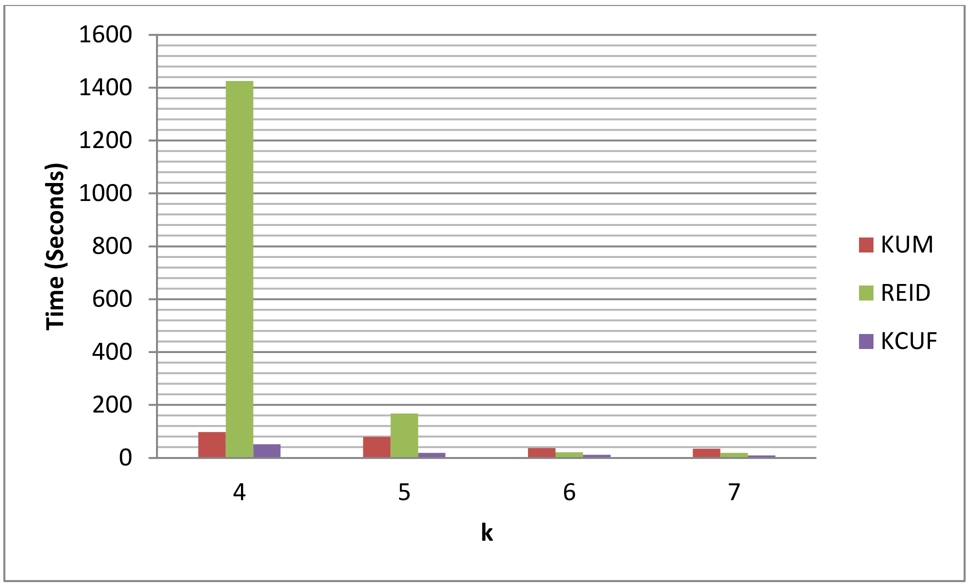


Figure 6.13 Time taken to detect communities in Amazon for different values of k

To summarize, number of communities detected by popular algorithms KUM and REID and number of communities detected by KCUF are found exactly same. Moreover the distribution of communities for different scales is also uniform. These two outcomes establish the correctness of KCUF. The time taken to detect the communities in the graph for different values of k have also been compared and it has been found that KCUF overtakes the two competitive algorithms for any value of k. Hence, it can be concluded that, KCUF is accurate and is a time efficient community detection algorithm which can handle simple as well as complex graphs.

6.1.5 Topical Interaction Network Builder (TINB)

TINB is tested over more than 100 topics fetching millions of pages. The topics are picked from Open Directory Project (ODP)[7]. ODP is a directory where URLs are maintained category wise. The directory is maintained manually, hence unbiased. The data fetched from ODP is in RDF format. The current work advocates the use the data and URLs in this format to initialize parameters for focused crawl. Few topics are arbitrarily decided by defining the parameters and few known seeds. The crawler produces a collection of pages fetched from WWW. Every page crawled by the crawler is part of the interaction network. The performance of the crawler is determined by two objectives of the crawler.

- *Does the crawler maintain focussed direction of crawl?*
  The pages need to be evaluated in terms of their relevance to the focus area.
- *Does the crawler produce real world network of web pages?*
  The objective of the TINB is to prepare an interaction network of web pages which is similar to any trivial social network and keep that network as focused as possible.

In order to test the performance TINB, following performance metrics are identified from the literature [9, 86]:

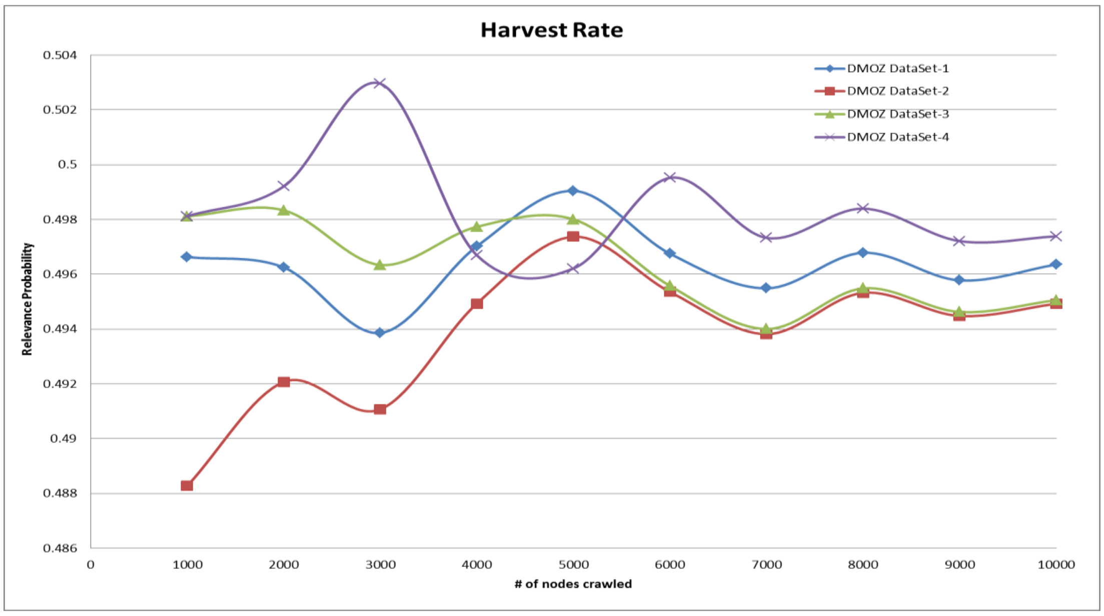
A) Harvest Rate

Harvest rate measures the fraction of the crawled pages which are relevant to the topic of focus. Any manual judgement of relevance is not feasibly possible to conduct on millions of pages. Suppose the crawler collects t pages during the crawl, harvest rate is defined as equation 6.5:

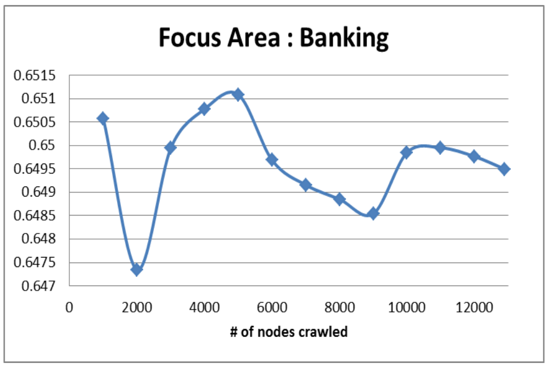
[7] dmoz-odp.org

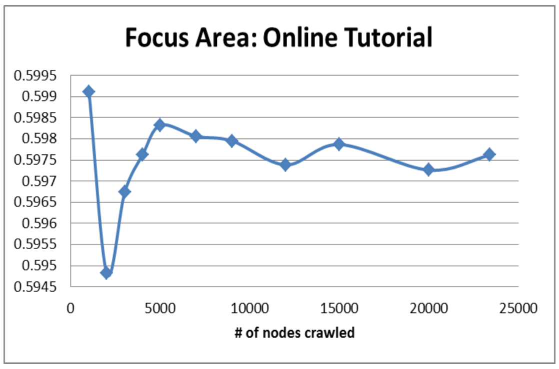
$$h = \frac{1}{t}\sum_{i=1}^{t} P(u_i) \qquad (6.5)$$

Where, $P(u_i)$ denotes relevance probability of i^{th} page represented by URL u_i. Relevance probability ranges from 0 to 1. 0 means the page is completely irrelevant to the focus area and 1 means the page satisfies every parameter of focus area.



(a)





(b) (c)

Figure 6.14 Average Harvest Rate. Vertical axis shows average relevance probability. (a) Topics picked from Open Directory Project, (b) Focus area is banking, (c) Focus area is online tutorial

Figure 6.14 shows harvest rate of crawls done on topics picked from Open Directory Project (ODP) and topics configured manually. Almost 50% harvest rate is maintained by the crawler in crawls for topics selected from ODP and 60% harvest rate is maintained in case of topics set manually.

B) Degree Distribution

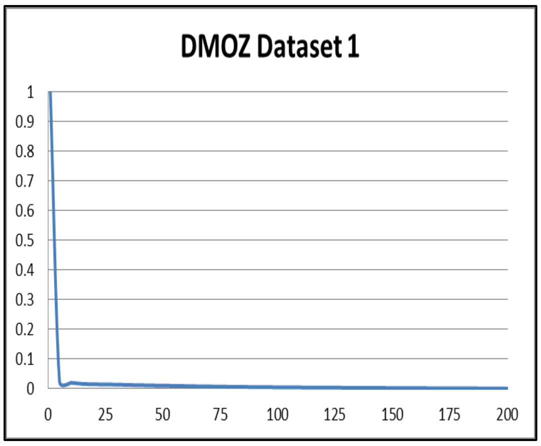
Degree of the vertex in a network is number of edges incident on it. Let us define P_k be the fraction of nodes having degree k. In other words it can be said as, P_k is the probability of any vertex chosen randomly which has degree k. A plot of P_k for any network gives distribution of degree. For any random network the degree distribution follows binomial or poison distributions [169 - 171]. But the real world networks are unlikely to show their degree distribution as Random networks. The degrees of vertices in real world networks are highly skewed towards right. There are two ways to plot degree distribution. One is to construct a histogram with exponentially growing bin sizes (e.g. 1, 2-3, 4-7, 8-15…). In this scheme the histogram is plotted on logarithmic scale and width of each bin remains constant. Another way is to plot cumulative distribution of degrees represented by following equation 6.6.

$$P_k = \sum_{k' \geq k} P_{k'} \tag{6.6}$$

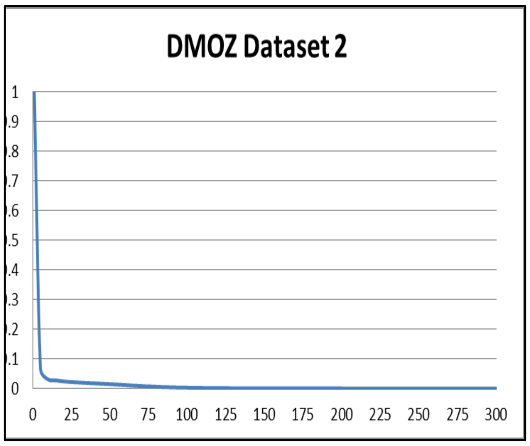
where, P_k is the probability of a node having degree greater than or equal to k. This method represents all the data whereas in binning, any difference between the values of data points is lost if they fall into the same bin. Social networks are also real world networks and they also showcase same degree distribution. Degree distribution of every real world network follows power law of degree distribution defined as equation 6.7.

$$P_k \propto k^{-\alpha} \tag{6.7}$$

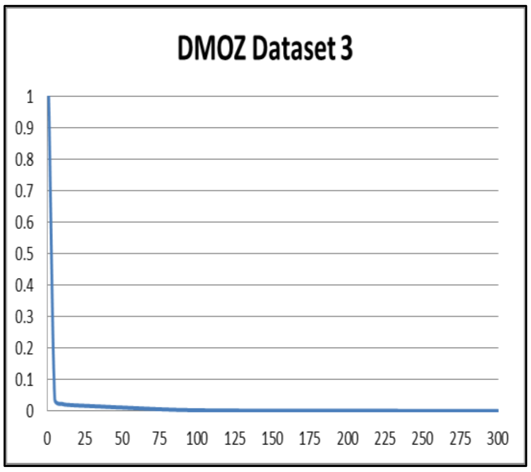
P_k is the probability of a node having degree k and α is some constant. Networks with power law degree distribution are referred as scale-free networks [172]. Figure 6.15 shows degree distribution of various interaction networks prepared by our crawler. The horizontal axis for each panel is vertex degree k and vertical axis is cumulative probability distribution of degrees, i.e., the fraction of vertices that have degree greater than or equal to k. It is evident that all distributions are right skewed and follow power law of degree distribution.



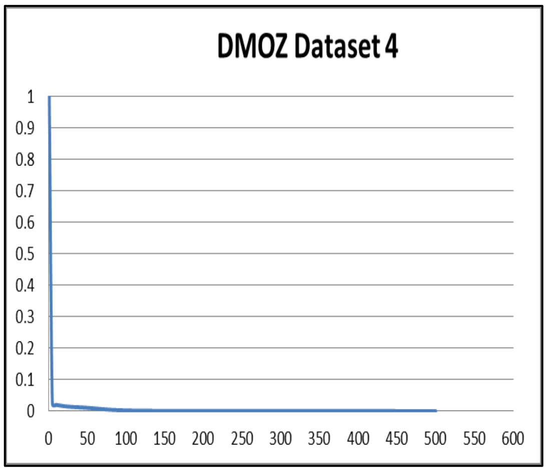
(a) Network of topics picked from Open Directory Project



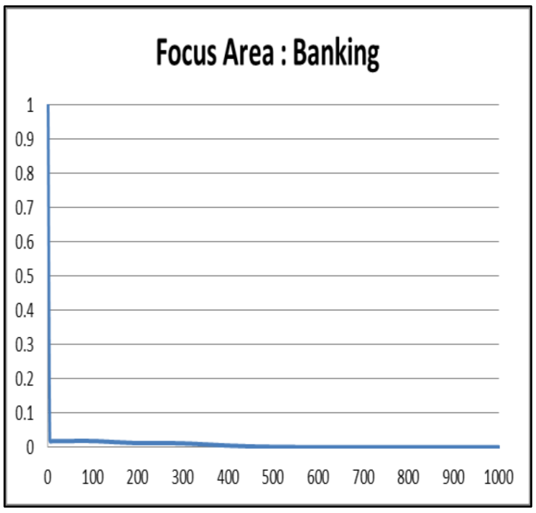
(b) Network of topics picked from Open Directory Project



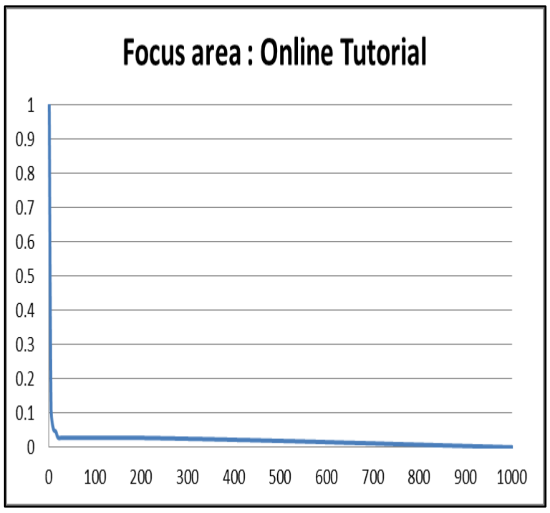
(c) Network of topics picked from Open Directory Project



(d) Network of topics picked from Open Directory Project

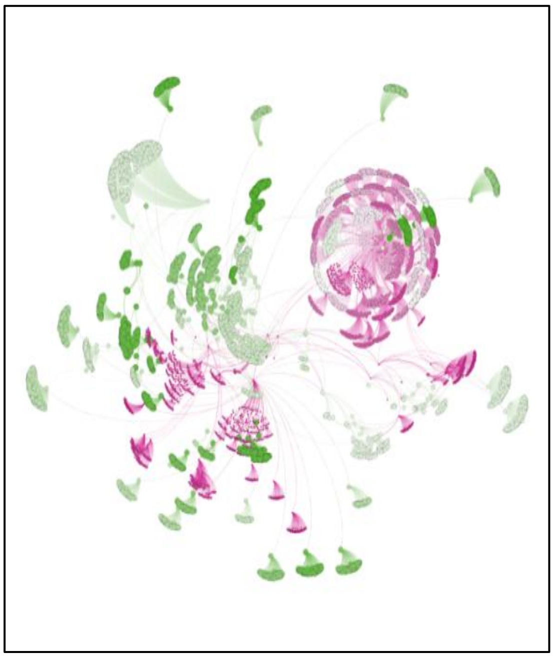


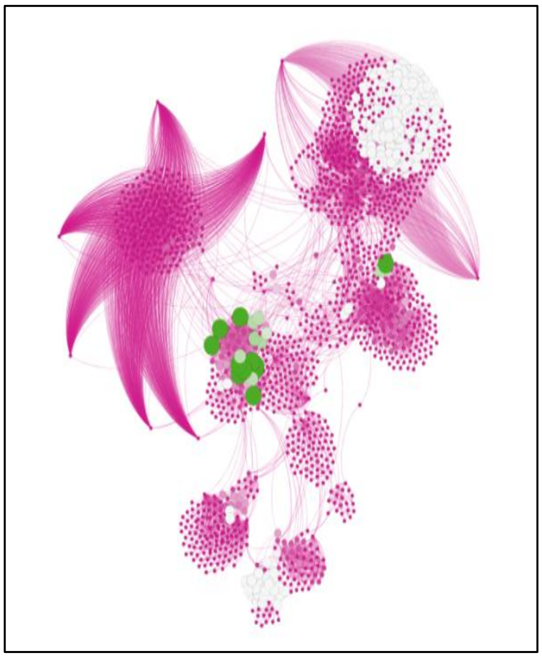
(e) Network of topic : Banking

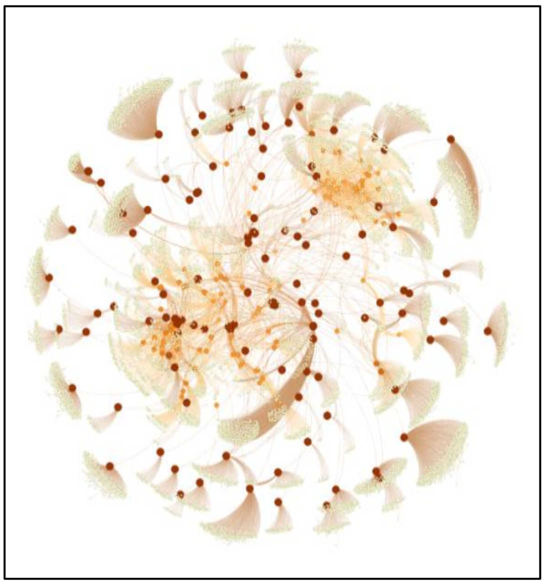


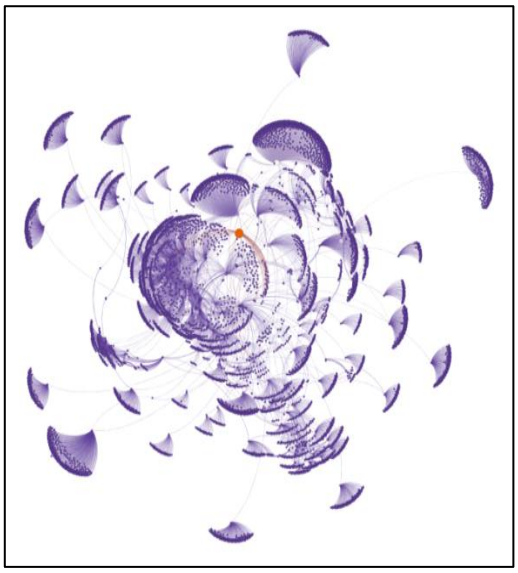
(f) Network of topic : Online Tutorial

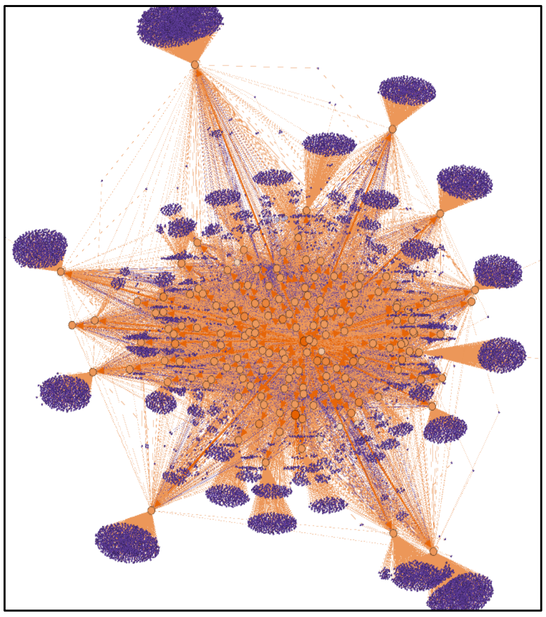
Figure 6.15 Cumulative degree distributions for six different networks.


(a) DMOZ Dataset-1, Modularity classes [95] in different color codes. 70 communities detected with modularity 0.906.

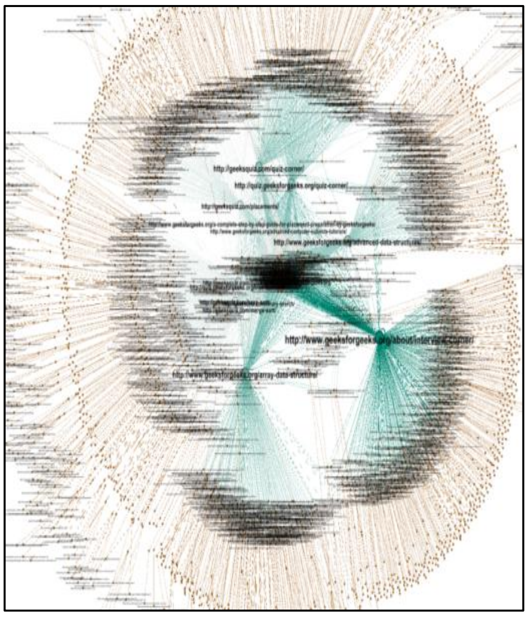

(b) DMOZ Dataset-2, Clustering Coefficient calculated through brute-force approach Clustering coefficient is 0.019.


(c) DMOZ Dataset-3, Closeness Centrality [173] is proportional to size of the node and intensity of color.


(d) DMOZ Dataset-4 Betweenness Centrality [173] is proportional to size of the nodes.



(e) Banking Dataset, Page Rank [55] of the web pages is proportional to size of the nodes.



(f) Online Tutorial, Degree Distribution is proportional to the size of the nodes.

Figure 6.16 Visualization of datasets with structural properties of the network.

Here, results of four crawls for topics selected from ODP and results of two crawls for topics configured manually are visualized using Gephi. Figure 6.16 shows visualization of these interaction networks. The visualization represents inter connection of the nodes graphically. The nodes represent certain web page (URL profile) and edges represent connection between these web pages. In all visualizations, one of several metrics of social graphs is highlighted with the help of colour coding or size of the nodes.

The structural properties of any social graphs help us to reach on various conclusions during analysis of social networks. In figure 6.16 (a – f) the interaction networks prepared from the datasets crawled by our crawler are visualized and their structural properties are exhibited, which establishes the usefulness of interaction networks in social research as any other trivial social network.

6.2 SUMMARY

The chapter discussed the implementation of proposed modules i.e. CCSC, FPBC, MCF, KCUF and TINB and the results thus obtained from testing these modules on standard datasets and performance metrics were presented. CCSC could successfully collect clustered view of the original network and it also outpaces the existing sampling techniques BFS, NBRW-rw and MHRW. FPBC could also effectively follow the foot prints and the deviation of samples in two crawls could well be compared. By observing the extent of deviation and frequency of FPBC iterations, evolution of dynamic social network could be captured. Further, results obtained via MCF indicated that it is a time efficient clique finding algorithm. KCUF deployed MCF and the results establish the fact that KCUF is able to detect correct number of communities with same distribution in the graph for any value of k in lesser time as compared to KUM and REID algorithms. This makes KCUF a better choice for community detection.

TINB is another significant contribution of this research work and the results reflect that TINB exhibits excellent harvest ratio for each interaction network and the interaction networks prepared have exactly same structural characteristics as does by any traditional social network. The results show that the SNA techniques and procedures are directly applicable on interaction networks prepared by TINB.

The results of this work are competitive and have an extra edge over existing works in the literature.

CHAPTER 7

CONCLUSIONS AND FUTURE SCOPE

7.1 CONCLUSIONS

The research began by exploring social network and its analysis. In depth study of related literature presented in chapter 2 indicated that SNA is a branch of social research and data analytics and very recently, SNA has gained tremendous popularity in various computation and research domains. Studies summarized that SNA has been equipped and enhanced with analytical procedures, tools and software. As the name indicated, SNA analyses social networks and identifies their structural characteristics to draw conclusions based on the application domain. One of the several reasons of SNA getting paramount importance are the business benefits that OSNs tycoons like Facebook, Twitter, Instagram, just to list a few, are gaining. With the inception of OSNs, a huge amount of social network data is easily available to conduct analysis and research. The most beautiful characteristic of OSNs is that they contain real time interaction data.

Social network analysis has emerged as a twin step process starting from collection of data from the social network leading to analysis of the data thus collected in the previous step. Owing to easy availability of social network data on internet and high computing devices, tools and softwares have been developed to collect social web data as well as for performing analysis of that data.

While the above process seemed to be a simple process initially, like any other existing solutions, it also embraced certain challenges. The very first challenge for social web data collection was gigantic size of the social networks. In fact, crawling a huge social network demanded a large amount of time and computing capacity. It was completely infeasible to collect complete social network. It seemed more feasible to collect a sample of the original social network instead of collecting complete social network. The sample had been considered to be a representative of the original social network and the results of SNA on the sample could hold on the original network too. However, this ideal condition was possible only if the sample possess same structural characteristics as does by the original social network.

Another challenge that emerged during studies was '*biasing*'. The social web crawlers tend to traverse similar kind of nodes. The sample created in this manner is not a good representative of original social network.

Further, dynamic behavior of social networks is another typical issue. The actors in the social network are prone to change their associations with other actors. They may remove old connection and may form new associations. Due to this dynamic behavior of social networks, the social network data collected in the repository soon got obsolete. In this case the results obtained from analysis of an obsolete social network data could not hold for the original social network.

The proposed framework "*An Adaptive Analyzer for Large Social Networks Using Distributed Crawling*" could address above highlighted challenges. The proposed framework comprises of various modules namely CCSC [41], FPBC [42], MCF [45], KCUF [44] and TINB [43] and the entire research had been implemented majorly in three phases.

In phase 1, CCSC performed the functionality of collecting sample of social network without getting biased towards higher degree nodes. FPBC which significantly contributed towards phase 2 of the work is actually an extension of CCSC and it could handle dynamic behavior of social network. Phase 3 majorly contributed MCF and KCUF where the former could find cliques and later could detect communities. MCF had been deployed as subroutine in KCUF. TINB is an additional module which could aid to phase 1 as objective of phase 1 was to make correct social data available for analysis. TINB represents non-social domains as social network which is suitable for every SNA technique.

Significant contributions made by this work are being presented in the next section.

7.2 SIGNIFICANT CONTRIBUTIONS

The work uniquely contributed 11 publications (10 published, 1 communicated) published in national and international journals and conferences of repute. Following is the summary of significant contributions made in support of this research work:

- An extensive literature survey was carried out to ensure the novelty and feasibility of work and the efforts are reflected through publication tilted as

"*Social Network Analysis: Hardly Easy*" published in *IEEE conference*. Another article titled *"A Walk Through Social Network Analysis: Opportunities, Limitations and Threats", published as book chapter indexed in Thomson Reuters Book Citation Index/Web of Science/ACM Digital Library).*

- CCSC successfully handles the problem of biasing and is able to target clustered view of the social networks. CCSC deploys a cluster coverage sampling algorithm and the authenticity of the same via article titled as "*Crawling Social Web with Cluster Coverage Sampling*" published in *Software Engineering. Advances in Intelligent Systems and Computing, vol 731, Springer*, indexed in *DBLP and SCOPUS*.

- FPBC successfully handles dynamic behaviour of social networks and is an extension of CCSC. It has been published in *SCOPUS indexed international journal International Journal of Pure and Applied Mathematics* with title "*Maintaining Freshness of Dynamic Social Network Data Using Foot-Prints Based Crawler".*

- Combating terrorism is a sensitive yet significant area of SNA application. An article titled "*Social Network Analysis as Counter Terrorism Tool"* published in *International Journal of Computer Science and Engineering* does an extensive exploration into the significant work done in this direction.

- A wide-ranging research survey was carried out to understand significant work done for community detection in social graphs and the challenges incorporated along with this. It is reflected in publications titled "*Study of Clique Based Community Detection Algorithms"* published in *International Journal of Computer Science and Engineering* and *"Social Graph Dynamism from Community Perspective"* published in *International Journal of Computer Networks and Applications*.

- Based on the challenges and feasibility assurance obtained from the above mentioned literature survey, a novel and time efficient k-clique based community detection algorithm KCUF was developed and is published as an article titled "*K-Clique Based Community Detection using Union Find - KCUF*" in *International Journal of Information Technology and Electrical Engineering*. It is worth mentioning that KCUF produced competitive results as compared to its competitors like KUM [132] and REID [136]. However; it outperformed the two in terms of time efficiency.

- KCUF is based on k-cliques and finding cliques in graph is a time taking process. A detailed review of clique finding algorithms in the literature and their performances and limitations have been examined and are reflected in article titled "*Finding Maximum Clique*" published in *National Conference on New Horizons in Technology for Sustainable Energy and Environment*.

- Based on the challenges and limitations in finding clique obtained from above mentioned study, a novel clique finding algorithm MCF has been proposed and published with title "*Maximum Clique Finder – MCF*" in *International Journal of Information Technology and Electrical Engineering*. MCF is deployed as subroutine in KCUF.

- Due to overwhelming popularity of SNA in various fields it seems promising if SNA techniques are utilized for non-social domains. An additional framework TINB has been developed to prepare an interaction network of web pages related to a specific topic. An article titled "*Building Interaction Network from WWW using Parameterized Crawler"* has been communicated for publication in *SCI indexed IEEE Transactions on Knowledge and Data Engineering*.

Although the research has made significant unique contributions, however; there is always the scope of further improvments. The next section focuses on future research directions in the field under consideration.

7.3 FUTURE SCOPE

Some of the functionalities can be further extended in proposed framework as follows:

A) Automatic Adaptation of Changes into SNA Results

In case of dynamic social network, FPBC is able to determine the freshness of the sample and if the sample is found to be obsolete then the sample in the dataset is updated with the latest sample. In this way the results obtained and conclusions drawn on obsolete sample become invalid. A mechanism can be developed by integrating FPBC with SNA technique in such a way that the changes identified by FPBC are directly updated in the dataset and the repercussions of structural changes in sample are injected in the already calculated results. This may save a huge amount of time and computation power as the results are not needed to be recalculated but they are adjusted according to the changes along with the dataset refreshment process.

B) Making Clique Finding and Community Detection Efficient for Gigantic Graphs

Finding maximal clique in a graph is NP-Hard problem. The algorithm proposed in this work is exact algorithm and tries to reduce the time by applying various pruning techniques. By the use of some heuristics the time can be further reduced with the compromise over the size of maximum clique found. Cliques are directly used in community detection. Therefore heuristic approach will make community detection process faster.

C) Extending KCUF for Dynamic Social Networks

KCUF is a time efficient static community detection algorithm and it beats other significant community detection algorithms based on k-clique with same accuracy. KCUF can be extended to work for dynamic social networks where temporal changes can be incorporated with results obtained in previous snapshot of the network. The idea is injecting the repercussions of temporal changes in previous results.

D) Intelligent SNA Framework

CCSC and FPBC use cluster coverage sampling methodology which is adaptive in nature. Once the crawling has started, no intervention is required and the crawler changes its direction autonomously. This leads to a possibility to design an intelligent framework for social network analysis consisting modules related to data gathering as well as analysis of that data working in sync. The data collection module takes feedback from analysis module and responds in terms of feedback from data collection module to analysis module. In this way a complete framework can be developed which functions on set parameters autonomously. The dataset is refreshed time to time and the analysis results are modified as per the changes automatically.

Undoubtedly, there are several other significant challenges in social network analysis which shall be answered by the researchers as per their interest. Hopefully, more efficient SNA tools and techniques shall be developed in future.

REFERENCES

[1] D. M. Boyd and N. B. Ellison, “Social Network Sites: Definition, History, and Scholarship”, in *Journal of Computer-Mediated Communication,* vol. 13, no. 1, Oct. 2007, pp.210-30.

[2] All twitter statistics at one place. Socialbakers. http://www.socialbakers.com/statistics/facebook/

[3] All facebook statistics at one place. Socialbakers. http://www.socialbakers.com/statistics/twitter/

[4] http://www.facebook.com

[5] http://www.flickr.com

[6] http://www.livejournal.com

[7] http://www.orkut.com

[8] http://www.linkedin.com

[9] M. Jamali, H. Abolhassani, "Different Aspects of Social Network Analysis," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2006, pp.66-72.

[10] S. Wasserman, and K. Faust, “Social Network Analysis, Methods and Applications”, *Cambridge University Press*, 1994.

[11] P. J. Carrington, J. Scott, and S. Wasserman, “Models and Methods in Social Network Analysis”, *Cambridge University Press*, 2005.

[12] A. Srivastava, Anuradha, D. J. Gupta, “Social Network Analysis: Hardly Easy”, in *IEEE International Conference on Reliability, Optimization and Information Technology*, 6, February 2014, pg. 128-135.

[13] Kneifer, J. Christopher, “A Comparison Study on Violent Video Games: Explained by the Gamers Themselves”, in *Graduate Theses and Dissertations,* 2014.

[14] W. Nooy, A. Mrvar, and V. Batagelj, “Exploratory Social Network Analysis with Pajek”, in *Cambridge University Press*, 2005.

[15] S. Ye, J. Lang, and F. Wu, “Crawling online social graphs,” in *Proceedings of International Asia-Pacific Web Conference*, 2010, pp. 236–242.

[16] Zhefeng Xio, Bo Liu, Huaping Hu, Tian Zhang, "Design and Implementation of Facebook Crawler Based on Interaction Simulation", in *Proceedings of*

IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 25-27 June 2012, pp.1109-1112.

[17] A. Saroop, A. Karnik, "Crawlers for social networks & structural analysis of Twitter", in *Proceedings of IEEE 5th International Conference on Internet Multimedia Systems Architecture and Application (IMSAA)* , 12-13 Dec. 2011, pp.1-8.

[18] M. E. J. Newman, “The Structure of Scientific Collaboration Networks”, in *proceedings of the National Academy of Sciences of the United States of America*, vol. 98, 2001, pp. 404-409.

[19] V. Krebs, “How to do Social Network Analysis”, [Online], Available: *http://www.orgnet.com/sna.html,* 2006.

[20] L. C. Freeman, “The Study of Social Networks”, [Online], Available: *http://www.insna.org/INSNA/na_inf.html,* 2002.

[21] J. Scott, “Social Network Analysis”, in *Newbury Park CA: Sage*, 1992.

[22] M. E. J. Newman, “The Structure and Function of Complex Networks”, in *SIAM Review*, vol. 45, 2003, p.167-256.

[23] R. Hanneman, and M. Riddle, “Introduction to Social Network Methods”, [Online textbook], Available: *http://www.faculty.ucr.edu/~hanneman/nettext/,* 2005.

[24] M. R. Garey and D. S. Johnson, “Computers and Intractability: A Guide to the Theory of NP Completeness”, in *New York, NY, USA: W. H. Freeman & Co.*, 1979.

[25] E. Kate and C. Inga, “Inside Social Network Analysis”, in *IBM Technical Report*, 2005.

[26] D. J. Watts, and S. H. Strogatz, “Collective Dynamics of ‘Small-World’Networks”, in *Nature*, vol. 393, 1998, pp. 440-442.

[27] S. Fortunato. “Community Detection in Graphs.” in *Physics Reports 486:3,* 2010, pp. 75–174.

[28] J. Chen, H. Zhang, Z. H. Guan, T. Li, "Epidemic spreading on networks with overlapping community structure," in *Physica A: Statistical Mechanics and its Applications*, Volume 391, Issue 4, 2012, pp. 1848-1854, ISSN 0378-4371.

[29] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. “Uncovering the Overlapping

Community Structure of Complex Networks in Nature and Society.” *Nature 435*, 2005, pp. 814–818.

[30] S. Sadi, S. Oguducu, and A. S. Uyar. “An Efficient Community Detection Method Using Paralle Clique Finding Ants”, in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2010, pp.1–7.

[31] A. P. Campbell, “Using LiveJournal for Authentic Communication in EFL Classes”, in *The Internet TESL Journal*, vol. 5, No. 9, 2004.

[32] http://www.myspace.com

[33] W. Aiello, F. Chung, L. Lu, “A Random Graph Model for Massive Graphs”, in *proceedings of the 32nd ACM Symposium on the Theory of Computing*, 2000, pp. 171-180.

[34] A. L. Barabasi, H. Jeong, Z. Neda, and E. Ravasz, A. Schubert, T. Vicsek, “Evolution of the Social Network of Scientific Collaborations”, in *Physica A: Statistical Mechanics and its Applications*, vol. 331, Issue 3-4, 2002, pp. 590-614.

[35] F. Liljeros, C. R. Edling, L. A. N. Amaral, H. E. Stanley, and Y. Aberg, “The Web of Human Sexual Contacts”, in *Nature*, vol. 411, 2001, p. 907-908.

[36] S. Redner, “How Popular is Your Paper? An Empirical Study of the Citation Distribution”, in *European Physical Journal B Condensed Matter and Complex Systems*, vol. 4, 1998, p 131-134.

[37] L. A. Adamic, “The Small World Web”, in *proceedings of the Third European Conference, ECDL’99 (Springer-Verlag, Berlin)* , 1999, pp. 443-452.

[38] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, “Graph Structure in the Web”, in *Computer Networks*, vol. 33, 2002, pp. 309-320.

[39] R. F. Cancho, and R. V. Sole, “The Small-World of Human Language”, in *Proceedings, Biological sciences* vol. 268, 1482, 2001, pp. 2261-2265.

[40] J. Leskovee, C. Faloutsos, “Sampling from Large Graphs”, in *KDD’06 Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery & Data-Mining*, 2006, pp. 631-636.

[41] Atul Srivastava, Anuradha, Dimple Juneja Gupta, “Crawling Social Web with Cluster Coverage Sampling”, In: *Hoda M., Chauhan N., Quadri S., Srivastava*

P. (eds) Software Engineering. Advances in Intelligent Systems and Computing, vol 731. Springer, Singapore, 2018, pp. 103-114.

[42] Atul Srivastava, Anuradha, Dimple Juneja Gupta, "Maintaining Freshness of Dynamic Social Network Data Using Foot-Prints Based Crawler", in *International Journal of Pure and Applied Mathematics*, Volume 120, issue 6, September, 2018, pp. 4693-4708.

[43] Atul Srivastava, Anuradha, Dimple Juneja Gupta, "Building Interaction Network from WWW using Parameterized Crawler", *Communicated to IEEE Transactions On Knowledge And Data Engineering*, 2019.

[44] Atul Srivastava, Anuradha, Dimple Juneja Gupta, "K-Clique Based Community Detection using Union Find KCUF", in *International Journal of Information Technology and Electrical Engineering*, Vol. 7 Issue 4, 2018, pp. 9-15, ISSN: 2306708X.

[45] Atul Srivastava, Anuradha, Dimple Juneja Gupta "Maximum Clique Finder", in *International Journal of Information Technology and Electrical Engineering*, Vol. 7 Issue 2, 2018, pp. 18-25, ISSN: 2306708X.

[46] S. Ressler, "Social Network Analysis as an Approach to Combat Terrorism: Past, Present, and Future Research", in *Homeland Security Affairs*, vol. II, Issue 2, 2006.

[47] D. M. Akbar Hussain. "Terrorist Networks Analysis through Argument Driven Hypotheses Model". In *Second International Conference on Availability, Reliability and Security (ARES'07)*. IEEE 0-7695-2775-2. 2007.

[48] J. Diesner, K. M. Carley, "Using network text analysis to detect the organizational structure of covert networks", in *Proceedings of the North American Association for Computational Social and Organizational Science (NAACSOS) Conference*, 2004.

[49] M. Tsvetovat, K. M. Carley, "On effectiveness of wiretap programs in mapping social networks", in *Computational and Mathematical Organization Theory*, vol. 13(1), 2006, pp. 63–87.

[50] Matthew J. Dombroski and Kathleen M. Carley, "NETEST: Estimating a Terrorist Network's Structure", in *11th European Intelligence and Security Informatics Conference (EISIC)*, Athens, 2011.

[51] Clifford Weinstein, William Campbell, Brian Delaney, Gerald O'Leary."

Modeling and Detection Techniques for Counter-Terror Social Network Analysis and Intent Recognition" *IEEE. 978-1-4244-2622-5,* 2009 . pp. 1-16

[52] M. Cheathan, K. Cleereman, "Application of Social Network Analysis to Collaborative Team Formation", in *International Symposium on Collaborative Technologies and Systems*, 2006, pp. 306-311.

[53] W. Almansoori, O. Zarour, T. N. Jarada, P. Karampales, J. Rokne, R. Alhajj, "Applications of Social Network Construction and Analysis in the Medical Referral Process", in proceedings of *IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC)* , 2011, pp.816-823.

[54] Wang Weiduo, Wu Bin, Zhang Zhonghui, "Website clustering from query graph using social network analysis", in proceedings of *IEEE International Conference on Emergency Management and Management Sciences (ICEMMS),* 2010, pp.439-442.

[55] S. Brin, and L. Page, "The anatomy of a large-scale hypertextual web search engine", in proceedings of the *7th World-Wide-Web Conference (WWW7),* Brisbane, Australia, April 1998.

[56] J. Klienburg, "Authoritative sources in a hyperlinked environment", in proceedings of *Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, Clifornia, Janual 1998.

[57] M. C. Deconta, L. J. Obrst and K. T. Smith, "The Semantic Web: A duide to future of XML, Web Services and Knowledge Management", in *Wiley Publishing Inc*. 2003.

[58] A. Swartz and J. Hedler, "The Semantic Web: A network of content for the digital city", in proceedings of *Second Annual Digital Cities Workshop,* Kyoto, Japan, October 2001.

[59] L. Golbeck, B. Parsia, and J. Hendler, "Trust network on the semantic web", in Proceedings of *Cooperative Intelligent Agents*, Helsinki, Finland, 2003.

[60] L. Ding, L. Zhou, and T. Finin, "Trust based knowledge outsourcing for semantic web agents", in Proceedings of *IEEE/WIC International Conference on Web Intelligence,* Halifax, Canada, Octuber 2003.

[61] Wei Kang, Wen-wuDuan, Lin Wen, "Research on emergency information management based on the social network analysis — A case analysis of panic buying of salt," in proceedings *of International Conference on Management*

Science and Engineering (ICMSE), 2011, pp.1302-1310.

[62] A. Zelenkauskaite, N. Bessis, S. Sotiriadis, E. Asimakopoulou, "Interconnectedness of Complex Systems of Internet of Things through Social Network Analysis for Disaster Management," in proceedings of *4th International Conference on Intelligent Networking and Collaborative Systems (INCoS)* , 2012, pp.503-508.

[63] Minas Gjoka, Maciej Kurant, Carter T. Butts, and Athina Markopoulou, "Practical Recommendations on Crawling Online Social Networks", in *IEEE Journal On Selected Areas In Communications,* Vol. 29, Issue no. 9, October 2011, pp. 1872-18792.

[64] C. Wilson, B. Boe, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao, "User Interactions in Social Networks and their Implications", in Proceedings of *ACM EuroSys,* 2009, pp. 205-218.

[65] S. H. Lee, P. -J. Kim, and H. Jeong, "Statistical Properties of Sampled Networks", in *Physical Review E,* Vol 73, Issue 1, 2006, pp. 016-102.

[66] Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong, "Analysis of Topological characteristics of Huge Online Social Networking Services", in *Proceedings of 16th international conference on World Wide Web,* 2007, pp. 835-844.

[67] M. Gjoka, M. Kurant, C. T Butts, and A. Markopoulou, "Walking in Facebook: A Case Study of Unbiased Sampling of OSNs", in *Proceedings of IEEE INFOCOM,* San Diego, CA, 2010, pp. 1-9.

[68] B. Ribeiro and D. Towsley, "Estimating and Sampling Graphs with Multidimensional Random Walks", in *proceedings of 10th ACM SIGCOMM conference on Internet measurement,* 2010, pp. 390-403.

[69] Minsu Cho, Jungmin Lee, and Kyoung Mu Lee, "Reweighted Random Walks for Graph Matching", ECCV 2010, Part V, LNCS 6315, pp. 492–505, 2010

[70] Chul-Ho Lee Xin Xu Do Young Eun, "Beyond Random Walk and Metropolis-Hastings Samplers: Why You Should Not Backtrack for Unbiased Graph Sampling", in *ACM SIGMETRICS Performance Evaluation Review,* Volume 40, Issue 1, June 2012, pp. 319-330.

[71] Rong-Hua Li, Jeffrey Xu Yu, Lu Qin , Rui Mao , and Tan Jin, "On Random Walk Based Graph Sampling", in *Proceedings of IEEE 31st International Conference on Data Engineering,* Seoul, 2015, pp. 927-938.

[72] Bruno Ribeiro, Pinghui Wang, Fabricio Murai, and Don Towsley, “Sampling Directed Graphs with Random Walks”, in *Proceedings IEEE INFOCOM,* Orlando, FL, 2012, pp. 1692-1700.

[73] Corlette D. and Shipman F, “Capturing On-line Social Network Link Dynamics using Event-Driven Sampling”, in *Proceedings of IEEE International Conference on Computational Science and Engineering*, Vancouver, BC, 2009, pp. 284-291.

[74] Minas Gjoka, Carter T. Butts, Maciej Kurant, and Athina Markopoulou, “Multigraph Sampling of Online Social Networks”, in IEEE Journal on Selected Areas in Communications, vol. 29, no. 9, October 2011, pp. 1893-1905.

[75] Tianyi Wang; Yang Chen; Zengbin Zhang; Tianyin Xu; Long Jin; Pan Hui; Beixing Deng; Xing Li, "Understanding Graph Sampling Algorithms for Social Network Analysis", 31st International Conference on Distributed Computing Systems Workshops (ICDCSW), 2011 , pp.123,128.

[76] T. Wang, Y. Chen, Z. Zhang, P. Sun, B. Deng, and X. Li, “Unbiased Sampling in Directed Social Graph”, In *ACM SIGCOMM Computer Communication Review*, Vol. 40, Issue 4, 2010, pp. 401-402.

[77] B. Ribeiro and D. Towsley, “Estimating and Sampling Graphs with Multidimensional Random Walks,” in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, 2010, pp. 390-403.

[78] Brandon Oselio and Alfred O. Hero, “Multi-Layer Graph Analysis for Dynamic Social Networks”, in *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 4, Aug. 2014, pp. 514-523.

[79] Leung A, Roven Lin JN, Szeto P. Implementation of a Focused Social Networking Crawler.

[80] Catanese, Salvatore A., Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. "Crawling facebook for social network analysis purposes", in *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, 2011, Article No. 52.

[81] Jiang Yifei, and Qin Lv., “Efficient Social Website Crawling Using Cluster Graph”, in *Efficient Social Website Crawling Using Cluster Graph*, 2009.

[82] O. A. McBryan, “Genvl and WWWW: Tools for Taming the Web” in

Computer Networks and ISDN Systems, Volume 27, Issue 2, 1994, pp. 308, ISSN 0169-7552.

[83] N. Craswell, D. Hawking, S. E. Robertson, “Effective Site Finding Using Link Anchor Information” in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 250-257.

[84] B. D. Davison, “Topical Locality in the Web”, in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, 2000, pp. 272-279.

[85] De Bra, P.M.E., Post, R.D.J., “Information Retrieval in the World Wide Web: Making Client-Based Searching Feasible”, in *Computer Networks and ISDN Systems,* Volume 27, Issue 2, 1994, Pages 183-192, ISSN 0169-7552.

[86] S. Chakrabarti, M. Van Den Berg, B. Dom, “Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery” in *Computer Networks*, Volume 31, Issues 11–16, 1999, pp. 1623-1640, ISSN 1389-1286.

[87] M. Iwazume, K. Shirakami, K. Hatadani, H. Takeda, T. Nishida, “IICA: An Ontology-Based Internet Navigation System” in *Proceedings AAAI-96 Workshop Internet Based Information Systems*, 1996.

[88] M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, S. Ur, “The Shark-Search Algorithm: An Application: Tailored Web Site Mapping”, in *Computer Networks and ISDN Systems*, Volume 30, Issues 1–7, 1998, pp. 317-326, ISSN 0169-7552.

[89] F. Menczer, G. Pant, M. Ruiz, P. Srinivasan, “Evaluating Topic-Driven Web Crawlers”, in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 241-249.

[90] M. Subramanyam, G. V. R. Phanindra, M. Tiwari, M. Jain, “Focused Crawling Using TFIDF Centroid”, in Hypertext Retrieval and Mining (CS610) class project, 2001.

[91] P. Bedi, A. Thukral, H Banati, “A multi-threaded semantic focused crawler”, in *Journal Of Computer Science And Technology*, Volume 27, Issue 6, November 2012, pp. 1233–1242.

[92] H. Dong, F. K. Hussain, “Self-Adaptive Semantic Focused Crawler for

Mining Services Information Discovery", in *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, 2014, pp. 1616-1626.

[93] YaJun Du, YuFeng Hai, ChunZhi Xie, XiaoMing Wang, "An approach for selecting seed URLs of focused crawler based on user-interest ontology", in *Applied Soft Computing*, Volume 14, Part C, 2014, pp. 663-676.

[94] S. Y. Yang, "A Focused Crawler with Ontology-Supported Website Models for Information Agents", in *Bellavista P., Chang RS., Chao HC., Lin SF., Sloot P.M.A. (eds) Advances in Grid and Pervasive Computing. GPC. Lecture Notes in Computer Science,* vol 6104. Springer, Berlin, Heidelberg, 2010.

[95] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks", in *Journal of Statistical Mechanics: Theory Experiment*, Vol. 2008, Issue 10, 2008, pp. P10008.

[96] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," Proc. Nat. Acad. Sci. USA, vol. 105, no. 2, pp. 1118–1123, 2008.

[97] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks", in Physical Review E, Vol. 76, Issue 3, 2007, pp. 036106:1-11.

[98] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," in Physical Review E, Vol. 74, Issue 3, 2006, pp. 036104.

[99] T. Aynaud and J.-L. Guillaume, "Static community detection algorithms for evolving networks," in *Proceedings IEEE 8th International Symposium on Modelling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, Avignon, 2010, pp. 513-519.

[100] M. E. J. Newman, "Modularity and community structure in networks," in *Proceedings Nat. Acad. Sci. USA*, vol. 103, no. 23, 2006, pp. 8577–8582.

[101] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proceedings ACM 19th Int. Conf. World Wide Web*, 2010, pp. 631–640.

[102] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: The state-of-the-art and comparative study," in *ACM Computing Surveys (CSUR),* Volume 45 Issue 4, August 2013, Article No. 43.

[103] X. Wen et al., “A maximal clique based multiobjective evolutionary algorithm for overlapping community detection,” in *IEEE Transactions on Evolutionary Computation,* Vol. 21, Issue 3, June 2017, pp. 363-377.

[104] W. Cui, Y. Xiao, H. Wang, and W. Wang, “Local search of communities in large graphs,” in *Proceedings of SIGMOD,* 2014, pp. 991–1002.

[105] Y. Wu, R. Jin, J. Li, and X. Zhang, “Robust local community detection: On free rider effect and its elimination,” in *Proceedings VLDB Endowment,* vol. 8, no. 7, 2015, pp. 798–809.

[106] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang, “Dense subgraphs with restrictions and applications to gene annotation graphs,” in *Research in Computational Molecular Biology. Berlin, Germany: Springer,* 2010, pp. 456-472.

[107] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarl, “Denser than the densest subgraph: Extracting optimal quasi-cliques with quality guarantees,” in *Proceedings SIGKDD,* 2013, pp. 104–112.

[108] M. Sozio and A. Gionis, “The community-search problem and how to plan a successful cocktail party,” in *Proceedings SIGKDD,* 2010, pp. 939–948.

[109] F. Luo, J. Z. Wang, and E. Promislow, “Exploring local community structures in large networks,” in *Proceedings IEEE/WIC/ACM International Conference on Web Intelligence, Hong Kong,* 2006, pp. 233-239..

[110] K. J. Lang and R. Andersen, “Finding dense and isolated submarkets in a sponsored search spending graph,” in *Proceedings CIKM,* 2007, pp. 613–622.

[111] R. Andersen, F. Chung, and K. Lang, “Local graph partitioning using pagerank vectors,” in *Proceedings FOCS,* 2006, pp. 475–486.

[112] Moon J and W Moser, “On cliques in graphs”, in *Israel Journal of Mathematics,* Vol. 3, Issue 1, 1965, pp. 23-28.

[113] M. E. J. Newman, M. Girvan, “Finding and evaluating community structure in networks”, in *Physical Review E,* Vol. 69, pp. 026113.

[114] J. Chen, O. R. Za¨ıane, R. Goebel, “Local community identification in social networks”, in *Proceedings of International Conference on Advances in Social Network Analysis and Mining, Athens,* 2009, pp. 237-242.

[115] S. White, P. Smyth, “A spectral clustering approach to finding communities in graph”, in *Proceedings of the 2005 SIAM International Conference on Data*

Mining, 2005, pp. 274-285.

[116] M. S. Handcock, A. E. Raftery, J. M. Tantrum, “Model-based clustering for social networks”, in *Journal of the Royal Statistical Society: Series A (Statistics in Society),* Vol. 170, 2007, pp. 301–354.

[117] E. M. Airoldi, D. M. Blei, S. E. Fienberg, E. P. Xing, “ixed membership stochastic blockmodels”, in *Journal of Machine Learning Research,* Vol. 9, 2008, 1981–2014.

[118] D. S. Choi, P. J. Wolfe, E. M. Airoldi, “Stochastic blockmodels with growing number of classes”, in *Biometrika,* Vol. 99, Issue 2, 2012, pp. 273-284.

[119] P. Doreian, V. Batagelj, A. Ferligoj, A. “Generalized blockmodeling of two-mode network data”, in *Social Networks,* Volume 26, Issue 1, 2004, pp. 29-53, ISSN 0378-8733..

[120] P. Doreian, V. Batagelj, A. Ferligoj, A. “Generalized Blockmodeling”, in *Cambridge University Press*, 2005.

[121] H. White, S. Boorman, R. Breiger, “Social structure from multiple networks: I. blockmodels of roles and positions”, in *American Journal of Sociology,* Vol. 81, 1976, pp. 730–80.

[122] J. Leskovec, J. Kleinberg, C. Faloutsos, “Graphs over time: densification laws, shrinking diameters and possible explanations”, in *Proceedings of the 11th ACM SIGKDD international conference on Knowledge Discovery in Data Mining,* 2005, pp. 177-187.

[123] L. Backstrom, D. Huttenlocher, J. Kleinberg, X. Lan, “Group formation in large social networks: membership, growth, and evolution”, in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery And Data Mining,* 2006, pp. 44-54.

[124] R. Kumar, J. Novak, A. Tomkins, (2006), “Structure and evolution of online social networks”, in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery And Data Mining,* 2006, pp. 611–617.

[125] Tanja Falkowski, Anja Barth, Myra Spiliopoulou, “Studying Community Dynamics with an Incremental Graph Mining Algorithm”, in *Proceedings of the Fourteenth Americas Conference on Information Systems,* 2008.

[126] M. Ester, H. P. Kriegel, J. Sander, X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, in *Proceedings*

of KDD, 1996, pp. 226-231.

[127] M. Ester, H. P. Kriegel, J. Sander, M. Wimmer, X. Xu, “Incremental Clustering for Mining in a Data Warehouse Environment”, in *Proceedings of 24th VLDB Conference,* 1998, pp. 323-333.

[128] M. Takaffoli, F. Sangi, J. Fagnan, O. R. Za¨ıane, “A framework for analyzing dynamic social networks”, in *Proceedings of 7th Conference on Applications of Social Network Analysis ASNA,* 2010.

[129] M. Takaffoli, F. Sangi, J. Fagnan, O. R. Za¨ıane, (2011) {Community Evolution Mining in Dynamic Social Networks”, in *Elsevier Procedia - Social and Behavioral Sciences*, 2011, pp. 49 – 58.

[130] J. Huang, B. Yang, D. Jin, Y. Yang, (2013) “Decentralized mining social network communities with agents”, in *Elsevier Mathematical and Computer Modelling*, Vol. 57, 2013, pp. 2998-3008.

[131] Imre Derenyi, Gergely Palla, and Tamas Vicsek, “Clique percolation in random networks”, in *Physical Review Letters,* Vol. 94, Issue 16, 2005, pp. 160202:1-4.

[132] Jussi M. Kumpula, Mikko Kivela, Kimmo Kaski, and Jari Saramaki, “A sequential algorithm for fast clique percolation”, in *Physical Review E,* Vol. 78, Issue 2, 2008, pp. 026109:1-7.

[133] Shi Xiaohong, Liang Luo, Wan Yan, Xu Jin, “A finding maximal clique algorithm for predicting loop of protein structure”, in *Applied Mathematics and Computation,* Vol. 180, 2006, pp. 676–682.

[134] Sumana Maity and Santanu Kumar Rath, “Extended Clique Percolation Method to Detect Overlapping Community Structure”, in *Proceedings of International Conference on Advances in Computing, Communications and Informatics (ICACCI),* 2014, pp. 31-37.

[135] Ali Varamesh, Mohammad Kazem Akbari, Mehdi Fereiduni, Saeed Sharifian, Alireza Bagheri, “Distributed Clique Percolation based Community Detection on Social Networks using MapReduce”, in *Proceedings of IEEE 5th Conference on Information and Knowledge Technology (IKT),* 2013, pp. 478-483.

[136] Fergal Reid, Aaron McDaid, Neil Hurley, “Percolation Computation in Complex Networks”, in *Proceedings of IEEE/ACM International Conference*

on Advances in Social Networks Analysis and Mining, 2012, pp. 274-281

[137] Fei Hao, Geyong Min, Zheng Pei, Doo-Soon Park, Laurence T. Yang, "K-Clique Community Detection in Social Networks Based on Formal Concept Analysis", in *IEEE Systems Journal*, Vol. 11, Issue 1, 2017, pp. 250-259.

[138] Shuai Wang, Chao Li, and Daquan Tang, "Service Identifying Based Weapon Systems Architecture Optimization with K-Clique Community Detection Algorithm", in *Proceedings of 3rd IEEE International Conference on Control Science and Systems Engineering (ICCSSE), Beijing*, 2017, pp. 656-660.

[139] Noha Alduaiji , Amitava Datta , and Jianxin Li, "Influence Propagation Model for Clique-Based Community Detection in Social Networks", in *IEEE Transactions on Computational Social Systems*, Vol. 5, Issue 2, 2018, pp. 563-575.

[140] T. Matsunaga, C. Yonemori, E. Tomita, and M. Muramatsu, "Clique-Based Data Mining for Related Genes in a Biomedical Database", in *BMC Bioinformatics*, Vol. 10, Issue 1, 2009, pp. 205.

[141] J. G. Augustson and J. Minker, "An Analysis of Some Graph Theoretical Cluster Techniques", in *Journal of the ACM*, Vol. 17, Issue 4, 1970, pp. 571–588.

[142] L. Wang, L. Zhou, J. Lu, and J. Yip, "An Order-Clique-Based Approach for Mining Maximal Colocations" in *Information Sciences*, Vol. 179, Issue 19, 2009, pp. 3370–3382.

[143] R. E. Bonner, "On Some Clustering Techniques", in *IBM Journal of Research and Development*, Vol. 8, Issue 1, 1964, pp. 22–32.

[144] R. Horaud and T. Skordas, "Stereo Correspondence Through Feature Grouping and Maximal Cliques", in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, Issue 11, 1989, pp. 1168–1180.

[145] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. D. Smith, "A New Table of Constant Weight Codes", in *IEEE Transactions on Information Theory*, Vol. 36, Issue 6, 1990, pp. 1334–1380.

[146] M. Pavan and M. Pelillo, "A New Graph-Theoretic Approach to Clustering and Segmentation", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, 2003, pp. 145–152.

[147] G. Gutin, J. Gross, J. Yellen, "Discrete Mathematics & Its Applications", in

Handbook of Graph Theory. CRC Press, 2004.

[148] P. M. Pardalos and J. Xue, “The Maximum Clique Problem”, in *Journal of Global Optimization*, Vol. 4, 1994, pp. 301–328.

[149] R. Carraghan and P. Pardalos, “An Exact Algorithm for the Maximum Clique Problem”, in *Operations Research Letters*, Vol. 9, Issue 6, 1990, pp. 375–382.

[150] P. R. J. Ostergard, “A Fast Algorithm for the Maximum Clique Problem”, in *Discrete Applied Mathematics*, Vol. 120, Issue 1-3, 2002, pp. 197–207.

[151] D. Johnson and M. A. Trick, “Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge”, in *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, 1993.

[152] E. Tomita, T. Seki, (2003) “An Efficient Branch-and-Bound Algorithm for Finding a Maximum Clique”, in *Calude C.S., Dinneen M.J., Vajnovszki V. (eds) Discrete Mathematics and Theoretical Computer Science, DMTCS, Springer*. *Lecture Notes in Computer Science*, Vol. 2731, 2003.

[153] E. Tomita, T. Kameda, "An Efficient Branch-and-bound Algorithm for Finding a Maximum Clique with Computational Experiments", in *Journal of Global Optimization*, volume 37, issue 1, 2007, pp. 95-111.

[154] E. Tomita, T. Kameda, "An Efficient Branch-and-bound Algorithm for Finding a Maximum Clique with Computational Experiments", in *Journal of Global Optimization*, volume 44, issue 2, 2009, pp. 311-311.

[155] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo, “The Maximum Clique Problem”, in *Handbook of Combinatorial Optimization*, 1999, pp. 1-74.

[156] P. San Segundo, D. Rodr´ıguez-Losada, and A. Jim´enez, “An Exact Bit-Parallel Algorithm for the Maximum Clique Problem”, in *Computers & Operations Research*, Vol. 38, Issue 2, 2011, pp. 571–581, ISSN 0305-0548.

[157] J. Konc and D. Janezic, “An Improved Branch and Bound Algorithm for the Maximum Clique Problem”, in *MATCH Communications in Mathematical Computer Chemistry*, Vol. 58, 2007, pp. 569–590.

[158] P. Prosser, “Exact Algorithms for Maximum Clique: A Computational Study”, in *Algorithms*, Vol. 5, Issue 4, 2012, pp. 545–587.

[159] Anusuiya Raj, Jean Francois Arvis, “How Social Connections and Business Ties Can Boost Trade: An Application of Social Network Analysis”, in *The*

Trace Post, 2014.

[160] Simon Cauchemez, Achuyt Bhattarai, Tiffany L. Marchbanks, Ryan P. Fagan, Stephen Ostroff, Neil M. Ferguson, David Swerdlow, The Pennsylvania H1N1 working group, "Role of social networks in shaping disease transmission during a community outbreak of 2009 H1N1 pandemic influenza", in Proceedings of the National Academy of Sciences, Vol. 108, Issue 7, 2011, pp. 2825-2830.

[161] V. Krebs, "Mapping networks of terrorist cells", in *Connections*, Vol. 24, Issue 3, 2002, pp. 43 – 52.

[162] S. Niskanen, P. R. J. Ostergard, "Cliquer user's guide, version 1.0", in *Technical Report T48, Communications Laboratory, Helsinki University of Technology, Espoo, Finland*, 2003.

[163] V. Batagelj, A. Mrvar, Pajek, Datasets. Available online (*http://vlado.fmf.uni-lj.si/pub/networks/data/*), 2006.

[164] M. E. J. Newman, "Co-authorship Networks and Patterns of Scientific Collaboration", in *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 101, 2004, pp. 5200–5205.

[165] D. Howe, "Foldoc: Free On-Line Dictionary of Computing." *Available online (http://foldoc.org/).*

[166] Google programming contest. *Available online (http://www.google.com/programming-contest/).*

[167] J. Leskovec, D. Huttenlocher, J. Kleinberg, "Predicting Positive and Negative Links in Online Social Networks", in *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*, 2010, pp. 641–650.

[168] S. R. Corman, T. Kuhn, R. D. Mcphee, K. J. Dooley, "Studying Complex Discursive Systems: Centering Resonance Analysis of Communication", in *Human Communication Research*, Vol. 28, Issue 2, 2002, pp. 157–206.

[169] Erdos, P., Renyi, A., 1959. On random graphs. Publ. Math. Debrecen, 6, pp. 290–297.

[170] Erdos, P., Renyi, A., 1960. On the evolution of random graphs. Magyar Tud. Akad. Mat. Kutato Int. Kozl., 5, pp. 17–61.

[171] Erdos, P., Renyi, A., 1961. On the strength of connectedness of a random graph. Acta Math. Acad. Sci. Hungar., 12, pp. 261–267.

[172] Barabasi, A. L., Albert, R., 1999. Emergence of scaling in random networks. Science, 286, pp. 509–512.

[173] Ulrik Brandes, 2001. A Faster Algorithm for Betweenness Centrality. Journal of Mathematical Sociology 25(2):163-177.

[174] Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, Koray Kavukcuoglu, "Interaction Networks for Learning about Objects, Relations and Physics", Published in NIPS 2016.

PROFILE OF RESEARCH SCHOLAR

Atul Srivastava did his M. Tech. in Information Technology from *J.C. Bose University of Science & Technology, YMCA,* Faridabad in 2012 and B. Tech. (Information Technology) from *Dr. A. P. J. Abdul Kalam Technical University (formerly Uttar Pradesh Technical University),* Lucknow in 2008. Mr. Atul has over 7 years of experience in teaching B. Tech. and M. Tech. courses. His areas of interest include Data Analytics, Distributed Computing, Theory of Computation, Search Engines and Social Network Analysis. He has published 24 research papers in various journals and conferences of international and national fame. He has worked as Assistant Professor in the department of Computer Science & Engineering at IET, Faridabad for 2 years and currently working in PSIT, Kanpur since 2014.

LIST OF PUBLICATIONS

List of Papers in Journals

S. No	**Title of the paper along with volume, Issue no., year of publication**	**Indexing**	**Publisher**	**Impact Factor**	**Refereed or Non-Refereed**	**Whether You paid any money or not for publication**	**Remarks**
1	A. Srivastava, A. Pillai, D. J. Gupta, "Social Network Analysis as Counter Terrorism Tool", International Journal of Computer Sciences and Engineering, Vol.6, Issue.9, pp.655-661, 2018.	UGC Approved	IJCSE	3.02	Yes	No	Published
2	Atul Srivastava, Anuradha, Dimple Juneja Gupta, “K-Clique Based Community Detection using Union Find KCUF”, International Journal of Information Technology and Electrical Engineering. Volume 7, Issue 4, pp. 9-15, August 2018.	UGC Approved	ITEE	3.75	YES	No	Published
3	A. Srivastava, A. Pillai, D. J. Gupta, "Study of Clique Based Community Detection Algorithms", International Journal of Computer Sciences and Engineering, Vol.6, Issue.9,	UGC Approved	IJCSE	3.02	Yes	No	Published

	pp.142-149, 2018.						
4	Atul Srivastava, Anuradha, Dimple Juneja Gupta, "Maintaining Freshness of Dynamic Social Network Data Using Foot-Prints Based Crawler", International Journal of Pure and Applied Mathematics, Volume 120, issue 6, pp. 4693-4708, September, 2018.	Scopus	Academic Publications	7.19	Yes	No	Published
5	Atul Srivastava, Anuradha, Dimple Juneja Gupta, "Maximum Clique Finder : MCF", International Journal of Information Technology and Electrical Engineering.. Volume 7, Issue 2, pp. 18-25, April 2018.	UGC Approved	ITEE	3.75	YES	No	Published
6	Atul Srivastava, Anuradha Pillai, Dimple Juneja Gupta "A Walk Through Social Network Analysis: Opportunities, Limitations and Threats", published in Natarajan Meghanathan. Ed. "Graph Theoretic Approaches for Analyzing Large-Scale Social Networks", published by IGI-Global, Hershey,	SCOPUS	IGI Global	0.78	Yes	No	Published

	USA, 2017. DOI: 10.4018/978-1-5225-2814-2.ch001.						
7	Atul Srivastava, Anuradha, Dimple Juneja Gupta, “Social Graph Dynamism from Community Perspective”, published in International Journal of Computer Networks and Applications, volume 5, special issue 2, pp. 31-36, ISSN: 2395-5317 December 2015.	UGC Approved	IJCNA	0.34	Yes	No	Published

List of Papers in Conferences

S. No.	Title of the paper along with volume, Issue no., year of publication	Indexing	Publisher	Remarks
1	Atul Srivastava, Anuradha, Dimple Juneja Gupta, Crawling Social Web with Cluster Coverage Sampling. In: Hoda M., Chauhan N., Quadri S., Srivastava P. (eds) Software Engineering. Advances in Intelligent Systems and Computing, vol 731. Springer, Singapore, 2018.	SCOPUS	Springer	Published
2	Atul Srivastava, Anuradha, Dimple Juneja Gupta, "Social Network Analysis: Hardly Easy", Proceedings of IEEE International Conference on Reliability Optimization & Information Technology, Feb, 2014.	SCOPUS	IEEE	Published
3	Atul Srivastava, Anuradha, Dimple Juneja Gupta, "Finding Maximum Clique", New Horizons in Technology for Sustainable Energy and Environment (NHTSEE 2017).	University	YMCA UST Faridabad	Published

List of Communicated Papers

S. No.	Title of the paper along with volume, Issue no., year of publication	Indexing	Publisher	Impact Factor	Referred or Non-Referred	Whether You paid any money or not for publication	Remarks
1	Atul Srivastava, Anuradha, Dimple Juneja Gupta, " Building Interaction Network from WWW using Parameterized Crawler", IEEE Transactions On Knowledge And Data Engineering. (SCI)	SCI	IEEE	2.77	YES	No	Communicated