Proximity Analysis of Social Network using Skip Graph

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

Submitted By
Amritpal Singh
(Roll No. 801131004)

Under the supervision of:
Dr. Shalini Batra
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

July 2013

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*Proximity Analysis of Social Network using Skip Graph*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Shalini Batra* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Amritpal Singh

(Amritpal Singh)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Shalini Batra)
Assistant Professor
Computer Science and Engineering Department,
Thapar University, Patiala

Countersigned by

(Dr. Maninder Singh)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

No volume of words is enough to express my gratitude towards my guide **Dr. Shalini Batra**, Department of Computer Science & Engineering, Thapar University, Patiala, who has been very concerned and has aided for all the materials essentials for the preparation of this thesis report. She has helped me to explore this vast topic in an organized manner and provided me all the ideas on how to work towards a research-oriented venture.

I am also thankful to **Dr. Maninder Singh**, Head of Computer Science & Engineering Department and **Mr. Karun Verma** , P.G. Coordinator, for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there at the need of hour and provided with all the help and facilities, which I required, for the completion of my thesis work.

Most importantly, I would like to thank my parents and the almighty for showing me the right direction out of the blue, to help me stay calm in the oddest of the times and keep moving even at times when there was no hope.

Amritpal Singh

Amritpal Singh

(801131004)

Abstract

With the enormous growth of Internet, Cloud Computing, etc. the world has become closer and faster and with the enormous growth of Social networks, Social Network Analysis (SNA) has come up as an important field for research. Social networks are represented as graphs and the fundamental component of SNA is the relationship defined by linkages among units or nodes in the network. Major concern for computer experts is how to store such enormous amount of data especially in form of graphs. Further, data structure used for storage of such type of data should provide efficient format for fast retrieval of data as and when required. Although adjacency matrix is an effective technique to store a graph having few or large number of nodes and vertices but when it comes to analysis of huge amount of data from site like face book or twitter, adjacency matrix are not sufficient. The intent of this thesis is to optimize the graph storage and mapping without using a large adjacency matrix to represent a large graph.

A special data structure Skip Graph, evolved from Skip list has been used as a replacement to a large adjacency matrix. Main advantage of skip graph is optimization in dynamic allocation of memory and fast search results with minimum computational cycles. It has been experimentally evaluated that the proposed approach significantly improves the space occupied by adjacency matrix and helps the graph to grow dynamically without prior knowledge of the size of network. Once the graph or social network is optimally stored, different type of analysis like proximity Analysis, Role Analysis, Centrality analysis and other information diffusion analysis can be done on stored network. Our major focus is on proximity analysis using some parameters which significantly affect the proximity of a node to its neighbors.

It has been experimentally proved that by using skip graph for social network analysis, access time for various operations such as insert, delete and traverse is significantly reduced and optimal storage, space utilization and retrieval can be achieved.

Table of Contents

List of Figures

Chapter 1
Introduction

1.1 Introduction

The enormous growth of Internet and the development of new applications and services have dramatically increased the number of users, resulting in increased memory size [1]. A social network is a set of relationships among interacting social entities called nodes. Although the growth in social networks is exponentially in past few years, no standard method has been designed for efficient mapping of a graph or social network onto a compatible data structure because of main two issues memory and computational time. As internet in itself is a social network graph comprises of nodes and edges (links) so with the increased size of graph day by day directly affect the storage being used, such as adjacency matrix [2, 3]. Data structures used for adjacency matrices usually have two components: an array that stores all the entries of the matrix and pointer to an array which would take care of increased size of entries in it [4].

1.2 Introduction to Social Network

Social network is a map of all of the relevant ties between all the nodes being studied, defined by graphs representing social relationships between people or organizations. Networks can range from small sociograms such as those introduced by Moreno [5] with only a few units to huge networks with practically billions of units, for example, a network of all computers connected to the Internet or a network of all people and their relationship or attributes. Units can be people, organizations, countries, words *etc.* and for each of these units there are a number of possible ties between pairs of these units. Commonly available networks are networks among people (friendship or communication), trade networks among organizations, countries, citation networks, genealogies, organic molecules in chemistry, ties among words in text, transportation networks *etc.*

Social networks are often displayed in a social network diagram, where nodes are the points and ties are the lines. Example includes internet network includes pages and links, email communication networks [6], instant messenger networks [7], mobile call networks [8], and

social networks [9], complex network, [10], biological networks, metabolic pathways, genetic regulatory networks, food web and neural networks etc. are also examined and demonstrate similar patterns [11]. Each node also called an actor or vertex in a graph represents an individual person or a group of person. An edge connecting two nodes called a tie represents relationship between the objects represented by these two nodes as shown in Figure 1.1.

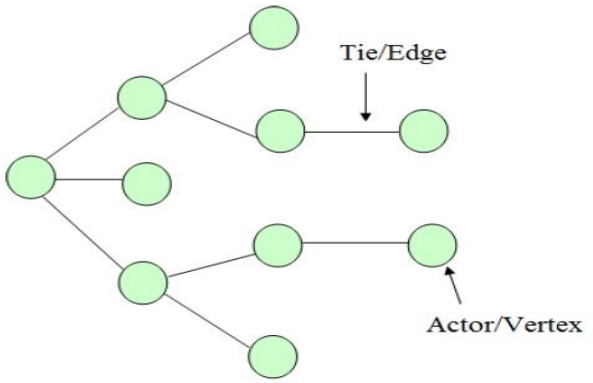

Figure 1.1: A Simple Network[12]

Social network analysis focuses on relationships between social entities. It offers the methodology to analyze social relations and it tells us how to conceptualize social networks. Social network analysts assume that interpersonal ties matter as do ties among organizations or countries, because they transmit behaviour, attitudes, information, or goods. These networks can also be used to measure social capital, the value that an individual gets from the social network. It is used widely in the social and behavioural sciences, as well as in political science, economics, organizational science, and industrial engineering. While people with similar attributes may behave similarly, explaining these similarities by pointing to common attributes misses the reality that individuals with common attributes often occupy similar positions in the social structure [5].Their similar outcomes are caused by the constraints, opportunities, and perceptions created by these similar network positions.

Using graphs to represent social data enables social analysts to completely and rigorously describe and analyze the structural information embedded in social relationships. In general social networks can be used to represent, identify, analyze and measure any type of correlations between any kind of entities (nodes) such as words, web pages, people,

organizations, animals, cells, computers, attribute and other information or knowledge processing entities.

1.3 Types of Social Network

Basically two kinds of social networks are investigated by social scientists:

- Egocentric networks
- Socio-centric networks.

Egocentric networks are connected with a single individual (the ego) and his or her social milieu expressed in formal and informal relations like kinship, friends, acquaintances, etc. Socio-centric networks on the other hand also called whole or complete networks are networks defined within a social system like the friendship relations in a classroom, a network of relations between workers and executives inside an organization; or relations between formal partner organizations. So depending upon the application used, a social network can be categorized into various types. As any real world problem can be mapped to a graph problem so a social network which is nothing but a graph having set of nodes and vertices can have various types. For example, cities can be treated as nodes and the path from one city to another can be treated as edge so the average flow of transport from one city to another can be treated as a weighted edge in a graph. Similarly a social network could contain multiple types of ties or the same type of ties with different weights. A network with multiple relations are called multi-relational network. So in short a social network can be weighted, unweighted, homogeneous, heterogeneous and many more. But in this thesis the main focus is given on unweighted and homogeneous relational network.

1.3.1 Example of Social Network

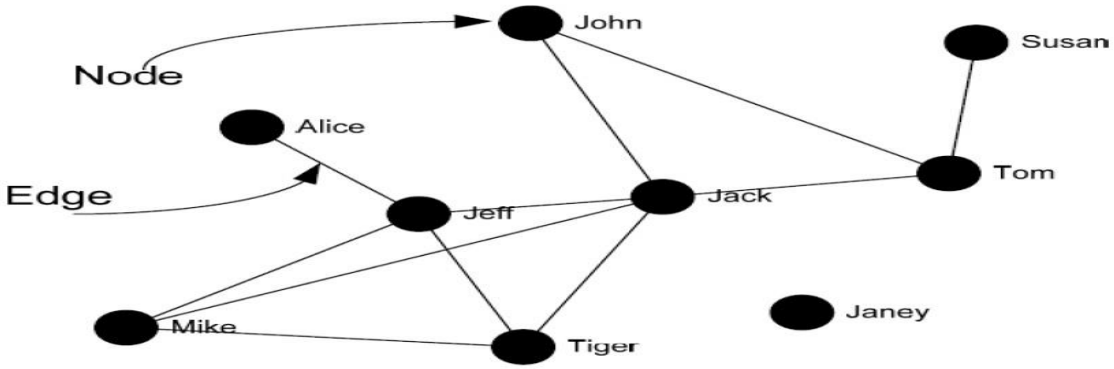

Figure 1.2: A Friendship Network [12]

In Figure 1.2 a real time example of a social networking site has been considered providing a small view of friendship network.

1.4 Basic Data Structures used to Store Graphical Data

1.4.1 Adjacency Matrix

The adjacency matrix of a finite graph G on n vertices is the $n \times n$ matrix where the non-diagonal entry a_{ij} is the number of edges from vertex i to vertex j, and the diagonal entry a_{ii}, depending on the convention, is either once or twice the number of edges (loops) from vertex i to itself. Undirected graphs often use the latter convention of counting loops twice, whereas directed graphs typically use the former convention. There exists a unique adjacency matrix for each isomorphism class of graphs (up to permuting rows and columns), and it is not the adjacency matrix of any other isomorphism class of graphs. In the special case of a finite simple graph, the adjacency matrix is a (0,1)-matrix with zeros on its diagonal. If the graph is undirected, the adjacency matrix is symmetric.[13].

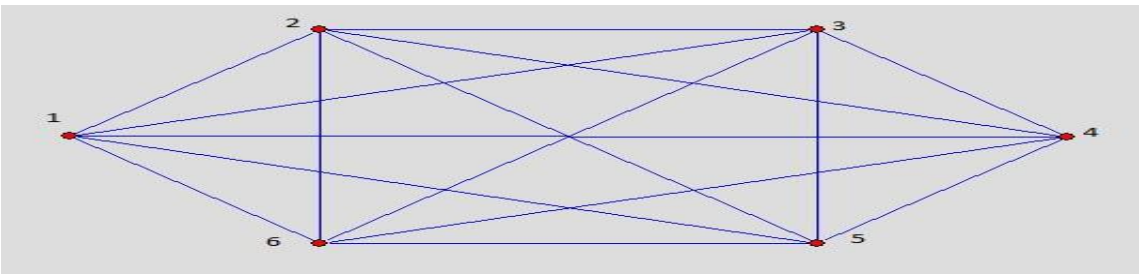

Figure 1.3: 6 node undirected network

	1	2	3	4	5	6
1.	0	1	1	1	1	1
2.	1	0	1	1	1	1
3.	1	1	0	1	1	1
4.	1	1	1	0	1	1
5.	1	1	1	1	0	1
6.	1	1	1	1	1	0

Figure 1.4: Adjacency Matrix representation of Fig 1.3 Network

The main drawback of this method is space complexity each new node added in graph increase size of Matrix by n^2 , where n is no. of nodes.

1.4.2 Adjacency List

An adjacency list represents a graph by associating each vertex in the graph with the collection of its neighbouring vertices or edges. [13] There are many variations of this basic idea, differing in the details of how they implement the association between vertices and collections, how they implement the collections, whether they include both vertices and edges or only vertices as first class objects, and what kinds of objects are used to represent the vertices and edges.

Edgeslist		Adjacent Edges				
1	--	2	3	4	5	6
2	--	3	4	5	6	
3	--	4	5	6		
4	--	5	6			
5	--	6				

Figure 1.5: Adjacency List representation of Fig 1.3 Network

1.5 Social Network Data Collection

The main objective of the data collection is to provide a data set that could help analyze the effects of Virtual Social Networks in the different aspects of social activities such as like their generation, their spatial distribution, similarities in behavior *etc.* [7]. However before building social networks the most important problem researchers have to face is how to acquire elementary data elements for building social networks. To get a complete and accurate description of interaction between individuals a lot of work has been done on social data gathering techniques focusing on how to identify the population, how to measure

relationships *etc.* Currently there are mainly two kinds of approaches for social network data gathering: elicitation and registration [8]. Elicitation acquires interaction information via the questionnaire/survey. Registration acquires interactions through extracting from registered information such as membership lists email records, author records of scientific articles *etc.*
In the early SNA research, questionnaire/survey was the method primarily used for collecting data which had high-labour cost. It required social scientists and network researchers to put lot of efforts to gather data for a network of even middle size (several thousands of nodes). This intensive labour cost considerably limited the size of networks to be studied. Through fast developments of computer technologies and universal applications of computers, automated data acquisition are found in most, if not all, fields. Interactions between objects can be stored as or implicated by electronic data *for e.g.* co-authorship of research articles can represent the collaboration between research scientists. If two authors appear on the same paper, there will be a collaboration connection between them. Through rapid growth of network size and data-sharing techniques, huge databases of social interactions have emerged in various fields. For instance, there are many large databases that maintain records of article authors in publications of miscellaneous research fields [9].

1.6 Social Network Analysis

Social Network Analysis (SNA) is the study of relations between individuals including the analysis of social structures, social position, role analysis, and many others [14, 15]. A graph G (V, E) is made of two sets (V: set of vertices E: set of edges) and analyzing the nature of relationships and connections between entities is a key towards understanding a variety of phenomena. A network can be constructed based on the response, with nodes representing individuals and edges the interaction between them. So any social network can be modelled in the form of graph in which the actor or people act as node or vertex of graph and the relationship between actors act as edge of graph. The scientific study of social networks has been ongoing for decades but in the past few years it has seen tremendous growth in its application and publicity. Social network analysis is an emerging area of research for computer scientists as it employs different 5 concepts from graph theory, probability, and statistics to solve problems in a wide range of disciplines. SNA is based on an assumption of the importance of relationships among interacting units or nodes. These relations defined by linkages among units or nodes are a fundamental component of SNA.

Mining problems for graph data includes applying techniques such as frequent pattern mining, clustering [5] and classification [6]. These methods are much more challenging in the graph domain because the structural nature of the data makes the intermediate representation and interpretability of the mining results much more challenging.

While people with similar attributes may behave similarly, explaining these similarities by putting their common attributes results together. Their similar outcomes are caused by the constraints, opportunities, and perceptions created by these similar network positions. Since social network analysis can be performed on many real world networks from different domains, a number of social network analysis methods have been designed for various tasks as per the analysis requirements.

According to Wasserman and Faust, social network data can be viewed as a social relational system characterized by a set of actors (nodes) and their social ties (edges) [8]. Additional information in the form of actor attribute variables or multiple relations (Analysis Factors) can be part of the social relational system.
Social network analysis aims at understanding the network structure by description visualization and statistical modelling where data consist of various elements. In the analysis of complete networks a distinction can be made between:

- Descriptive methods, through graphical representations
- Analysis procedures often based on a of the adjacency matrix
- Statistical models based on probability distributions.

Visualization by displaying a sociogram as well as a summary of graph theoretical concepts provides a first description of social network data. Wasserman and Faust stated that network analysis is based on the assumption of the importance of relationships (relations) among the interacting units [8].

1.7 Social Network Analysis Components

In recent times, the computer revolution has provided scholars with a huge amount of data and computational resources to process and analyze these data. For the analysis of a social network the main component that must be studied out is Community detection also known as clustering. Communities, also called clusters or modules, are groups of vertices which probably share common properties and/or play similar roles within the graph.

Society offers a wide variety of possible group organizations: families, working and friendship circles, villages, towns, nations. The diffusion of Internet has also led to the creation of virtual groups that live on the Web, like online communities. Normally data for huge graphs is stored in the form of adjacency matrix. One of the major problems associated with the use of adjacency matrix is that it has static array allocation and fixed entries for increased size of network creates problem during insertion. Secondly, dynamic array allocation requires more time to create a new array of increased size and then move the entries from previous array to new array and then deallocating the previous array [7, 8]. Finally, although the adjacency matrix for a graph is sparse, every null entry would take space. For dynamically increased network and efficient storage optimization [9, 10, 11, 12] this thesis proposes the use of Skip graph, an advanced version of linked list type data structure Skip list, to store a graph with its mapping information in an efficient manner. By using Skip Graph to store the problems of memory and dynamic allocation to new nodes in a graph are solved efficiently.

1.7.1 Social Network Analysis tasks

i. Centrality analysis: aims to identify the “most important” actors in a social network. Centrality is a measure to calibrate the “importance” of an actor. This helps to understand the social influence and power in a network.

ii. Community detection: Actors in a social network form groups. This task identifies these communities through the study of network structures and topology.

iii. Position/Role analysis: identifies the role associated with different actors during network interaction. It serves as the bridge between two groups.

iv. Network modelling: attempts to simulate the real-world network via simple mechanisms such that the patterns presented in large-scale complex networks can be captured. Information diffusion studies how the information propagates in a network.

v. Information diffusion: also facilitates the understanding the cultural dynamics, and infection blocking.

vi. Network classification and outlier detection: Some actors are labelled with certain information. For instance, in a network with some terrorists identified, is it possible to infer other people who are likely to be terrorists by leveraging the social network information.

vii. Viral marketing and link prediction: The modelling of the information diffusion process, in conjunction with centrality analysis and communities, can help achieve more cost-effective viral marketing. That is, only a small set of users are selected for marketing. Hopefully, their adoption can influence other members in the network, so the benefit is maximized.

1.8 Graph Clustering

Graph clustering is the process of organizing objects into groups whose members are similar in some way. A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters [14, 15]. If there is a friendship network then through clustering it would be easy to find out the related group of people and also community detection becomes very easy. Similarly if there is a transportation network then using clustering it becomes very easy to analyze the traffic between various cities and using centrality property [16] of clustering, possible actions can be taken to improve the transportation network. Figure 1.6 shows a schematic example of a graph with three communities.

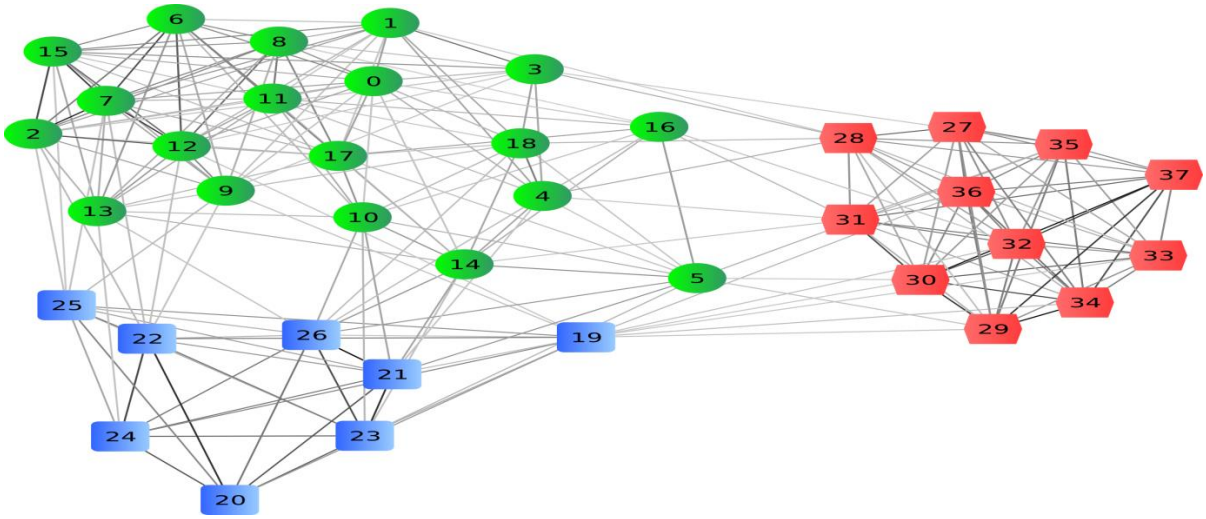

Figure 1.6: A Random Clustered Graph[12]

1.9 Social Network Analysis Applications

Social network analysis techniques can be applied to study structures of any types of interactions/relationships between any kinds of entities. From late 1970s, SNA techniques have gained massive attentions considerable developments, and successful applications in broad fields [14]. In companies and government agencies, there is lot of information sharing between workers. Using SNA tools on collaboration and/or information-sharing networks, managers can easily find the "important person" and build appropriate management strategies to improve efficiency. Combating terrorism is another field where SNA techniques have important and successful applications. Terrorist organizations have special structures on recruitment, evolution and radical ideas diffusion . SNA tools can be used to identify these unique organization structures and provide critical information for terrorist detection and terrorism prediction.

Social Network Analysis techniques also have been successfully applied in epidemiology. A lot of researchers try to analyze the spread of diseases based on the interactions between people. A SNA researcher, Valdis Krebs, listed a number of recent successful applications of SNA in [17].

A selected set of applications include:

- Identify these unique organization structures and provide critical information for terrorist detection and terrorism prediction.
- Examining a network of farm animals to analyze how disease spreads from one cow to another .
- Discovering emergent communities of interest amongst faculty at various universities .
- Revealing cross-border knowledge flows based on research publications .
- Determining influential journalists and analysts in the IT industry.
- Map executive's personal network based on email flows.
- Discovering the network of Innovators in a regional economy .

1.10 Structure of the Thesis

The rest of the thesis is organized in the following order:

Chapter 2 – Literature Review: This chapter reviews social network, Skip List and Skip Graph data structures for storage and mapping of clustering for analyzing social networks according to analysis factors.

Chapter 3 - Problem Statement: It states the problem and provides the methodology used to solve it.

Chapter 4 - Social Network Analysis Using Skip Graph: This chapter gives a detailed introduction about Skip List and Skip Graph data structure used to optimize graph storage and algorithm used for graph clustering.

Chapter 5 - Implementation Details: It includes the experiment performed using a real time example and the results achieved.

Chapter 6 - Conclusion and Future Scope: It concludes the thesis and provides suggestions for future work. Thesis concludes with references and list of publications.

Chapter 2
Literature Review

The origin of graph theory dates back to Euler's solution of the puzzle of Konigsberg's bridges in 1736 [18]. Since then a lot has been learned about graphs and their mathematical properties [19]. In the 20th century they have also become extremely useful as the representation of a wide variety of systems in different areas. Biological, social, technological, and information networks can be studied as graphs, and graph analysis has become crucial to understand the features of these systems. For instance, social network analysis started in the 1930's and has become one of the most important topics in sociology [20, 21].

Social network analysis goes back to Jacob L. Moreno and his psychodrama studies in the 1930's [17]. He was among the first that operationalized the concept of social network and represented interpersonal relations in small groups using graphical methods in order to visualize channels of information. This visual device was called sociograph and the branch that describes and analyzes this kind of network configurations was denominated sociometry. By the end of the 1940's important advances were made in the research of the structural properties of networks.

The scientific study of social networks has been ongoing for decades but in the past few years it has seen great growth in its application and publicity. Social network analysis is an emerging area of research for computer scientists as it employs different concepts from graph theory, probability, and statistics to solve problems in a wide range of disciplines. Similarly storage optimization has been expanding in all directions at an astonishing rate during the last few decades. New algorithmic and theoretical techniques have been developed, the diffusion into other disciplines has proceeded at a rapid pace, and knowledge of all aspects of the field has grown even more profound.

As research in this area grew, network analysis was distinguished from traditional social science by the dyadic nature of the standard data set. These dyadic attributes (social relation) may be represented in matrix from by a square 1-mode matrix. But the data in traditional social science are represented as 2-mode matrices. However network analysis is not completely divorced from traditional social science and often has occasion to collect

and analyze 2—mode matrices. Some of the methods developed in network analysis are used in analyzing non-network data.

At the same time, one of the most striking trends in optimization is the constantly increasing emphasis on the interdisciplinary nature of the field. Optimization today is a basic research tool in all areas of engineering, medicine, and the sciences. The decision-making tools based on optimization procedures are successfully applied in a wide range of practical problems arising in virtually any sphere of human activity. Optimization is in memory usage and computational speed is always main concern of research in every field.

Depending on the nature of the problem, different techniques can be used to formulate and solve a typical optimization storage problem. Various data structure such as Treaps, Skip list, Skip Graphs deals with optimization problems, in which the objective and constraints can be formulated using only functions that are linear with respect to the decision variables. In nonlinear optimization, one deals with optimizing a nonlinear function over a feasible domain described by a set of, in general, nonlinear functions. The pioneering works on the gradient projection method by L.S. Lasdon [23] and J. Aspens [4] generated a great deal of research enthusiasm in the area of various data structures , resulting in a number of new techniques for solving large-scale dynamic graphs.

In many optimization problems such as cluster mapping and storage, as well as other applications, the input data, such as demand or cost, are stochastic. In addition to the difficulties encountered in deterministic optimization problems, the stochastic problems introduce the additional challenge of dealing with uncertainties. To handle such problems, one needs to utilize probabilistic methods alongside optimization techniques. This led to the development of a new area called Skip Graph data structure, whose objective is to provide tools to help design and control stochastic systems with the goal of optimizing their performance. Here main concern in optimization is dynamic allocation of memory, fast search results with minimum computational cycles.

There are many circumstances in which binary relations are defined between pairs of objects in sociology. For example, there are social relations between people in business, there are trading relations between firms, and in design there are functional dependencies between components. In all of these situations, the clustering of objects into densely interconnected blocks discovers the actual structure of the system. Since social network analysis can be performed on many real world networks from different domains a number of social network analysis methods have been designed for various tasks like Clustering,

Role Analysis , Proximity , Community Detection , Link Prediction and Information Diffusion .

Although the enormous growth of Internet, Social Network, Cloud Computing, etc. has brought the world closer and faster, major concern for computer experts is how to store such enormous amount of data especially in form of graphs. Further, data structure used for storage of such type of data should provide efficient format for fast retrieval of data as and when required. Although adjacency matrix is an effective technique to represent a graph having few or large number of nodes and vertices but when it comes to analysis of huge amount of data from site likes like face book or twitter, adjacency matrix cannot do this. In this thesis we provides a special kind of data structure, skip graph which can be efficiently used for storing such type of data resulting in optimal storage, space utilization and retrieval.

Skip List

A skip list [3] is an ordered data structure based on a succession of linked lists with geometrically decreasing numbers of items. The deterministic versions of skip list have guaranteed properties whereas randomized skip lists only offer high probability performance. This height (Hn) is the maximum length of a search path for any key from the top of the skip list. Devroye proved that this height Hn is of order (log n) [24].

Skip Graph

Skip graphs are a novel distributed data structure, based on skip lists, provide the full functionality of a balanced tree in a distributed system where elements are stored in separate nodes that may fall at any time as described in [3]. In a skip graph, the whole data structure can be distributed among a large number of nodes, and the structure provides good load balancing and fault tolerance properties. They are composed of tower of increasingly refined linked lists in various levels, each one with no head and doubly linked.

Since Skip graphs have functionality best suitable for P2P distributed environments [25] they perform queries based on key ordering, improving on existing search tools that provide only hash table functionality.

As per analysis done by James and Shah on skip lists or other tree data structures, skip graphs are highly resilient, tolerating a large fraction of failed nodes without losing connectivity [2,3]. In addition, constructing, inserting new elements, searching a skip graph and detecting and repairing errors in the data structure introduced by node failures can be done using simple and straightforward algorithms.

Application Areas of skip graphs

During past years interesting variants of skip graphs have been studied, like skip nets [26], skip webs [1] or rainbow skip graphs [25]. Because of its good results Skip graph is successfully implemented in other research area's .

- Due to the rise in popularity of peer-to-peer systems dynamic overlay networks have recently received a lot of attention [27]. An overlay network is a logical network formed by its participants over a wired or wireless routing network. The number of computers and users in such a network may reach millions. Therefore, research in this area has focused on improving scalability and efficiency of overlay networks. Two usual optimization properties are the speed of searching for items in the network and the speed of topology updates. Another popular parameter is expansion.

- In open peer-to-peer systems [participants may frequently enter and leave the overlay network either voluntarily or due to failures [27,28]. In a peer-to-peer system of large scale, faults and inconsistencies are the norm rather than as an exception. Moreover, interaction of such faults may leave the system in an unpredictable, possibly system-wide failure. Hence, overlay networks require mechanisms that continuously counter such disturbances. Paper [27] shows the successful implementation of Skip Graph in such problem.

- Hammurabi , Cristina shows that Skip graphs show better results in multithreaded multiprocessor or distributed systems , where many entities access the shared pool concurrently [25]. In such cases manual control of concurrency is required in all operations, and use of Skip graph show good results.

- The results shown by James , Wieder represent that high probability in a skip graph provide quality and more connectivity in expanding networks [26]. So these properties help them to construct unstructured P2P system, making it a good candidate for a hybrid P2P system.

- Cloud computing has evolved as a popular computing environment [24]. In data centric applications hosted on the cloud, data is accessed and updated in a purely distributed manner. The distributed data structures used for dynamic

storage of the data for such applications require two fundamental qualities, authentication and persistence, which are not completely met by existing distributed data structures. Authentication is a crucial requirement for data structures used on the cloud, as users need to be convinced about the validity of the data they receive. Moreover the data structure has to be persistent, so that changes can be made to the data structure without losing old data, or old versions of the data structure, which may be required by different users in the distributed environment. In [24] authors have shown that use of Skip graph in all basic models perform better from many existing models.

- The problems of Web Services where main issue is to support the reuse and interoperation of software components on the web are receiving ever increasing interests from e-commerce, science, and research communities across different areas. A fundamental problem of Web Services is service discovery. Web Services discovery is the process of finding a Web Service that is capable to deliver a particular service, or integrating several Web Services in order to achieve a particular goal by means like IR (Information Retrieval). In all alternatives Skip graph shows the better results for this problem [29].

- Skip graph shows the properties of Fast queries and updates and they also support for ordered data [31]. These feature allows for a richer set of queries than a simple dictionary that can only answer membership queries, including those arising in DNA databases, location-based services, and prefix searches for file names or data titles and helpful in communication by fast message exchange.

Benefits of Skip Graph

Correctness under concurrency

Both insertion and deletion can be comfortably done on skip graph and search operations eventually find their target node or correctly report that it is not present in the skip graph. So any search operation can be linearized with respect to insertion and deletion. In effect, the skip graph inherits the atomicity properties of its bottom layer, with upper layers serving only to provide increased efficiency.

Fault Tolerance

James and Udi describe some of the fault tolerance properties of a skip graph [4]. Fault tolerance of related data structures, such as augmented versions of linked lists and binary trees, has been well-studied by Munro and Poblete [31]. The main question is how many nodes can be separated from the primary component by the failure of other nodes, as this determines the size of the surviving skip graph after the repair mechanism finishes. It has been clearly proved that even a worst-case choice of failures by an adversary can do only limited damage to the structure of the skip graph. With high probability, a skip graph with n nodes has an tQ(1/logn) expansion ratio, implying that at most *O(f* log n) nodes can be separated .

Random failures

For random failures, the situation appears even more promising, experimental results presented in [4,26,28] show that for a reasonably large skip graph nearly all nodes remain in the primary component until about two-thirds of the nodes fail, and that it is possible to make searches highly resilient to failure even without using the repair mechanism by use of redundant links.

Fast Search and Fault Tolerance

The average search in skip graph involves only O(logn) nodes that most searches succeed as long as the proportion of failed nodes is substantially less than O(logn) [1,28,31] . By detecting failures locally and using additional redundant edges, one can make searches highly tolerant to small numbers of random faults. In general, results cannot make as strong guarantees as those provided by data structures based on explicit use of expanders [25,26], but this is compensated for by the simplicity of skip graphs and the existence of good distributed mechanisms for constructing and repairing them.

Load balancing

In addition to fault-tolerance, a skip graph provides a limited form of load balancing, by smoothing out hot spots caused by popular search targets. The guarantees that a skip graph makes in this case are similar to the guarantees made for survivability. Just as an element stored at a particular node will not survive the loss of that node or its neighbours in the graph, many searches directed at a particular element will lead to high load on the node that stores it and on nodes likely to be on a search path. However, James has shown that this effect drops off rapidly with distance elements that are far away from a popular target in the bottom-level list produce little additional load on average [4]. Further author has provided two characterizations of this result. The first shows that the probability that a particular search uses a node between the source and target drops off inversely with the

distance from the node to the target. This fact is not necessarily reassuring to heavily-loaded nodes. Since the probability averages over all choices of membership vectors, it may be that some particularly unlucky node finds itself with a membership vector that puts it on nearly every search path to some very popular target. Second characterization addresses load balancing issue by showing that most of the load-spreading effects are the result of assuming a random membership vector for the source of the search.

Low hitting times

Random walks on expanders done in [26] have the property of hitting a large set of nodes fast and with high probability. This can be used for a variety of applications such as load balancing, gathering statistics on the nodes of the skip graph and for finding highly replicated fact is not necessarily reassuring to heavily-loaded nodes. Since the probability averages over all choices of membership vectors, it may be that some particularly unlucky node finds itself with a membership vector that puts it on nearly every search path to some very popular target. Our second characterization addresses this issue by showing that most of the load-spreading effects are the result of assuming a random membership vector for the source of the search.

After going through various research proposals in the area of analyzing social network, it was realized that community detection or clustering can be used for analyzing complex data as well as dynamic scalable graph as it is based on the idea that units in a network can be grouped according to the extent to which they are equivalent under some meaningful definition of equivalence. To store a dynamic scalable graph and for optimal storage of communities, Skip Graphs evolved from Skip list are one of the best models to use.

Chapter 3

Problem Statement

3.1 Problem Definition

In SNA, statistical analysis of relational data is derived using various social networks modeling techniques which facilitates an awareness and understanding of the connections among people, whether they are political leaders, specific groups or cliques in an organization and using graphs to represent social data enables social analyses to completely and rigorously describe and analyze structural information embedded in the social relationships. In general, social network can be used to represent, identify and measure any type of correlations between any kind of entities such as words, web, pages, people, organization, animal, cells, computers, attributes and other information and knowledge processing entities. Major focus of this thesis is to provide an efficient data structure for storage and retrieval of graph data and find proximity of a node with its neighbouring node using various parameters.

- A Distributed data structure Skip Graph, advanced version of Skip list evolved from linked list data structure has been used to optimally store a large and scalable graph.
- Once the storage is done effectively, graph analysis is done to identify various parameter which play an important role in identifying the proximity of node with its neighbours.
- An algorithm has been designed to find proximity of a node with neighbours stored using skip graphs.

3.2 Methodology

The step-by-step methodology to be followed in storing, designing and analyzing of a social network is given below:

- Design of friendship network any graph in Pajek, a software tool for Automatic Network Analysis.
- Selection of appropriate parameters for SNA as per the application requirements i.e., identity whether a parameter is directly or indirectly linked with the scenario.

- Design an algorithm for proximity analysis.
- Implement the designed algorithm in programming languages like C or C++ and analyse the result achieved.

Chapter 4

Social Network Analysis Using Skip Graph

4.1 Introduction to Social Network Analysis

Social network is a network of all of the relevant edges between all the nodes being studied, defined by graphs representing social relationships between people or organizations [27]. Using graphs to represent social data enables social analysts to store completely and rigorously all information related to each and every node. To store the data in graphs a fast and memory efficient data structure is required . Data structures like adjacency matrix and list have been frequently used to store the graphical data of small size but as network grows the size of these data structure grows order of n^2 times. In such situations where memory consumption is huge and node size is dynamic, a data structure is required which can handle big data efficiently. In this thesis skip graph is proposed as one of the methods for efficient storage and retrieval of data for SNA.

4.2 Introduction to Skip Graph

Skip graphs are novel distributed data structures, based on skip lists, that has several benefits: Resource location and dynamic node addition and deletion can be done in logarithmic time, and each node in a skip graph requires only logarithmic space to store information about its neighbors. More importantly, there is no hashing of the resource keys, so related resources are present near each other in a skip graph. This may be useful for certain applications such as prefetching of Web pages, enhanced browsing, and efficient searching. Skip graphs also support complex queries such as range queries, i.e., locating resources whose keys lie within a certain specified range. Skip graphs are resilient to node failures: a skip graph tolerates removal of a large fraction of its nodes chosen at random without becoming disconnected, and even the loss of an O(1/log n) fraction of the nodes chosen by an adversary still leaves most of the nodes in the largest surviving component. Skip graphs can also be constructed without knowledge of the total number of nodes in advance.

4.3 Graph Storage using Skip Graph

Let G be a graph comprises of two main components (V, E) where V contains a set of vertices of graph and E contains a set of edges of graph. To store the whole graph corresponds to its mapping information consider a graph having few nodes as shown in Figure 4.1.

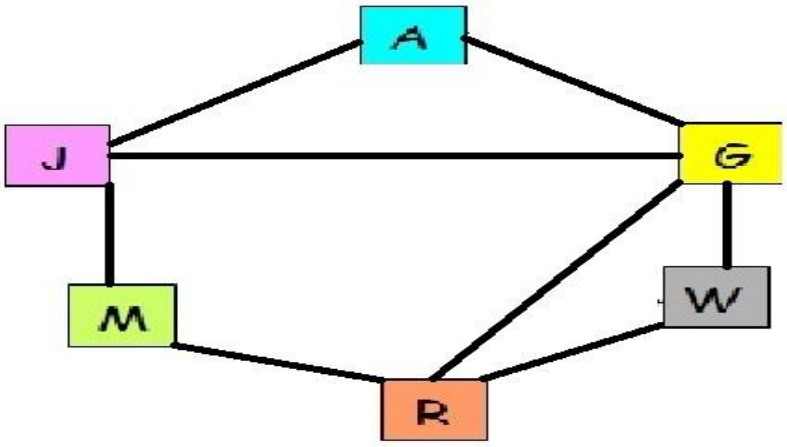

Figure 4.1: A Random Graph (Network)

The corresponding adjacency matrix for the graph in Figure 4.1 is:

	A	**J**	**M**	**R**	**W**	**G**
A	0	1	0	0	0	1
J	1	0	1	0	0	1
M	0	1	0	1	0	0
R	0	0	1	0	1	1
W	0	0	0	1	0	1
G	1	1	0	1	1	0

Figure 4.2: Adjacency Matrix for Figure 4.1

If the given adjacency matrix is used to store and map the graph information the storage schema used is not scalable for dynamic graph approach. But as the applications are increasing day by day where graph has a strong impact to represent a problem domain, using static allocation will not serve the purpose and in fact may cause problems during dynamic allocation.

Using a skip Graph to store the graph in figure 4.1 the output achieved is:

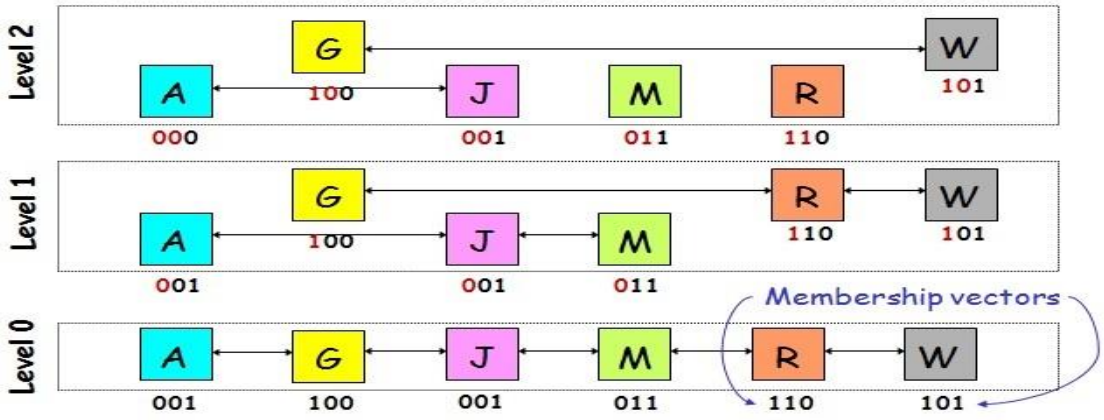

Figure 4.3: Skip graph for Figure 4.1 graph [31]

4.3.1 Notations and Structure of Skip Graph node

This section describes the search operation in a skip graph, structure of node of a skip graph for performing further operations. For simplicity, the key of a node (e.g. x.key) are referred with the same notation (e.g. x) as the node itself. It will be clear from the context whether a node or its key is referred. In the Skip Graph, the pointer to x's successor and predecessor at level "l" are denoted as x.neighbor[R][l] and x.neighbor[L][l] respectively. x_{Rl} is defined formally to be the value of x.neighbor[R][l], if x.neighbor[R][l] is a non-nil pointer to a non-faulty node, and @(some constant) otherwise and x_{Ll} is defined similarly. The variables stored at each node are summarized in Figure 4.4. It is assumed that the pointer variables are maintained by an underlying deadlock-free implementation of a distributed sorted doubly-linked list, as in [12]. Further, it is assumed that operations on this doubly-linked list can be treated as atomic, and that it supports search, insert, and delete operations that each take O(k) time and O(k) messages starting from a node k hops away from its target. As per the assumption, operations on different doubly-linked lists can be performed in parallel without interference.

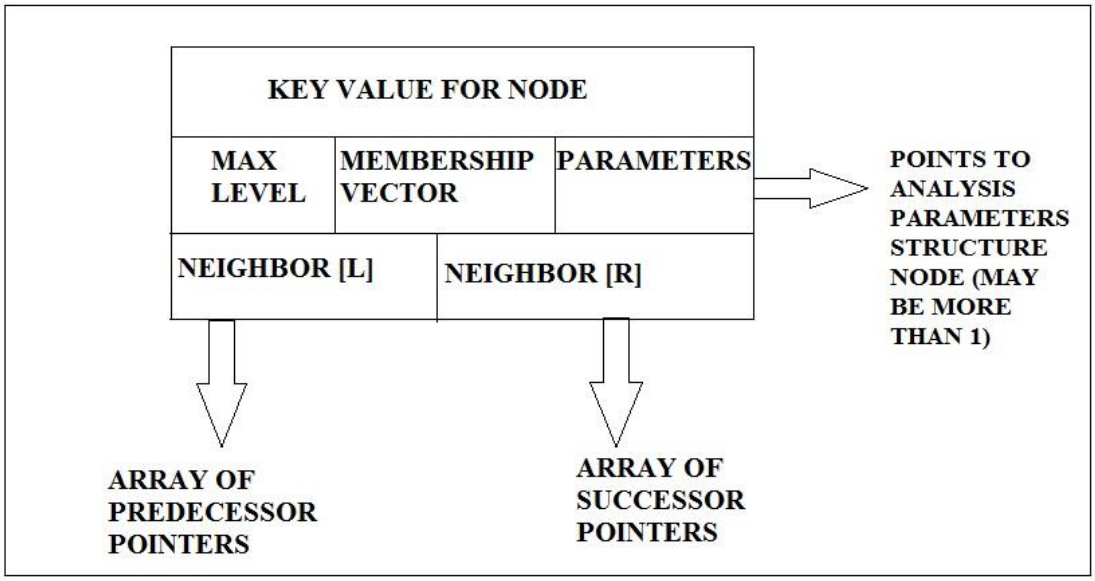

Figure 4.4: Structure of Skip Graph Node [4]

4.4 Operations on Skip Graph

Various operations in a Skip Graph can be performed using standard algorithms, such as insert new node, delete a node, traverse a node. For creation of Skip Graph, 'n' nodes would recall insert Algorithm:

4.4.1 Algorithm for insertion of a node

To insert a node in a Skip Graph:

Input:

- Key value of node as "key"
- Values of Left and Right neighbor for lateral updates.

Output:

- Insert the new node to its appropriate position in Skip Graph.

Insert a node In Skip Graph

A new node 'u' knows some introducing node 'v' in the network that will help it to join the network. Node 'u' inserts itself in one linked list at each level till it finds itself in a singleton list at the topmost level. The insert operation consists of two stages:

- Node 'u' starts a search for itself from 'v' to find its neighbours at level 0, and links to them.

- Node ‘u’ finds the closest nodes ‘s’ and ‘y’ at each level L >= 0, s < u < y, such that m(u) |(L+ 1) = m(s) | (L + 1) = m(y) |(L + 1), if they exist, and links to them at level L + 1.

Because each existing node ‘v’ does not require $m(v)_{L+1}$ unless there exists another node ‘u’ such that m(v) | (L + 1) = m(u) | (L + 1), it can delay determining its value until a new node arrives asking for its value; thus at any given time only a finite prefix of the membership vector of any node needs to be generated.

Detailed pseudo code for the insert operation is given in Algorithm 1.

Algorithm 1: insert for new node u

```
1  execute search from introducer to find max s < u
2  ℓ ← 0
3  while true do
4  |  insert u in list at level ℓ starting from s
5  |  scan backwards at level ℓ to find s' such that m(s') ⌈ (ℓ+1) = m(u) ⌈ (ℓ+1)
6  |  if no such s' exists then
7  |  |  exit
8  |  else
9  |  |  s ← s'
10 |  |  ℓ ← ℓ+1
```

Figure 4.5: Insertion algorithm [4]

Example of Insertion operation in Skip Graph

Considering the network from figure 4.1

Let us assume that a new node ‘J’ is to be added to the skip graph.

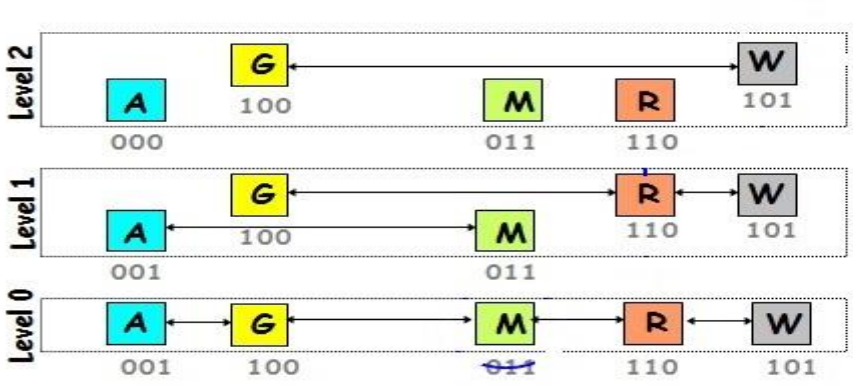

Figure 4.6: Skip Graph for Insertion

Steps followed for insertion of Node ‘J’ are:

- First find the exact location for insertion of new node by traversing the graph or searching for same node so that one gets address of the node storing the largest key less than the search key .

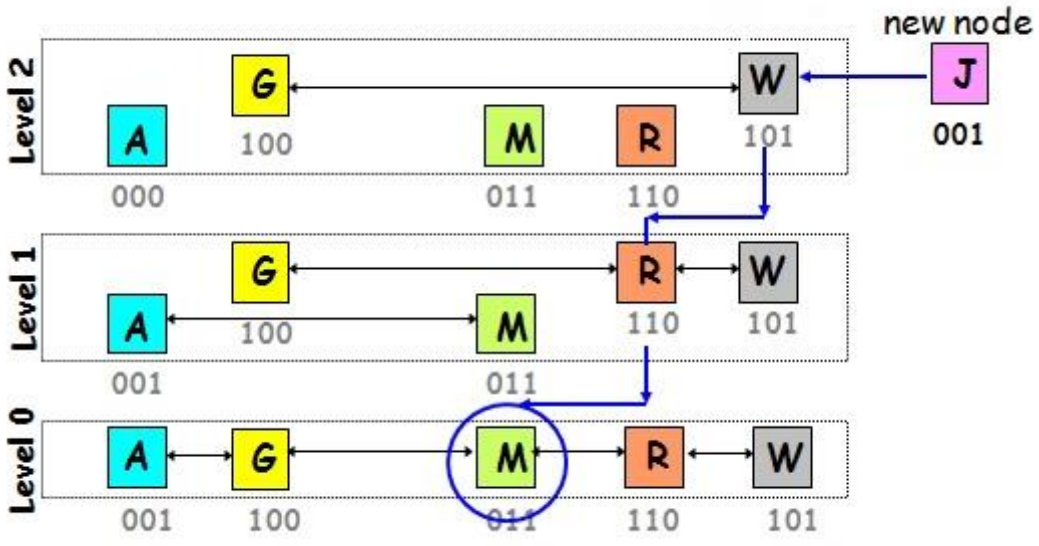

Figure 4.7: insertion of new node ‘J’ [30]

- Then all the links and membership vectors are updated as per the new node’s requirements (Figure 4.8).

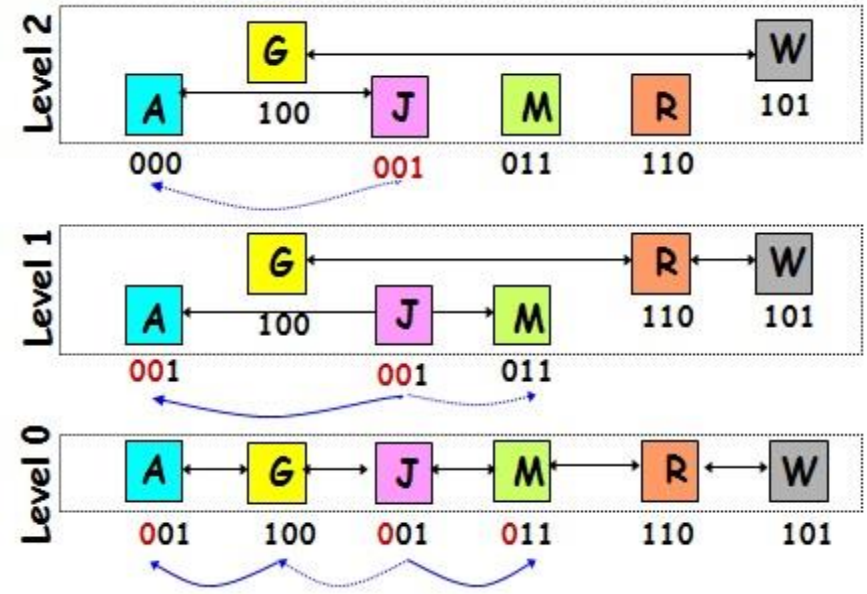

Figure 4.8: Graph after Inserting node ‘J’ [30]

4.4.2 Algorithm for searching of a node

To search a node in a Skip Graph:

Input:

- Key value of node as "key"

Output:

- If it exists the address of the node storing the search key is returned, But if node does not exists then address of the node storing the largest key less than the search key is returned.

Search Algorithm for a node 'v' in Skip Graph

- The search starts at the topmost level of the node seeking a key and it proceeds along each level without overshooting the key, continuing at a lower level if required, until it reaches level 0.

Algorithm 2: search for node v

```
1  upon receiving ⟨searchOp, startNode, searchKey, level⟩:
   // If the current key is the search key, return
2  if (v.key = searchKey) then
3  |  send ⟨foundOp, v⟩ to startNode
   // Coming from left, current key is less than search key
4  if (v.key < searchKey) then
   |  // Go one level down till a right neighbor with key smaller than the search key is reached
5  |  while level ≥ 0 do
6  |  |  if ((v.neighbor[R][level].key ≤ searchKey) then
7  |  |  |  send ⟨searchOp, startNode, searchKey, level⟩ to v.neighbor[R][level]
8  |  |  |  break
9  |  |  else level←level-1
   // Coming from right, current key is greater than search key
10 else
11 |  while level ≥ 0 do
   |  |  // Go one level down till a left neighbor with key larger than the search key is reached
12 |  |  if ((v.neighbor[L][level]).key ≥ searchKey) then
13 |  |  |  send ⟨searchOp, startNode, searchKey, level⟩ to v.neighbor[L][level]
14 |  |  |  break
15 |  |  else level←level-1
   // Reached the bottom-most level, key is not found
16 if (level < 0) then
17 |  send ⟨notFoundOp, v⟩ to startNode
```

Figure 4.9 Searching algorithm [4]

Example of Search operation in Skip Graph

Again considering the Figure 4.1

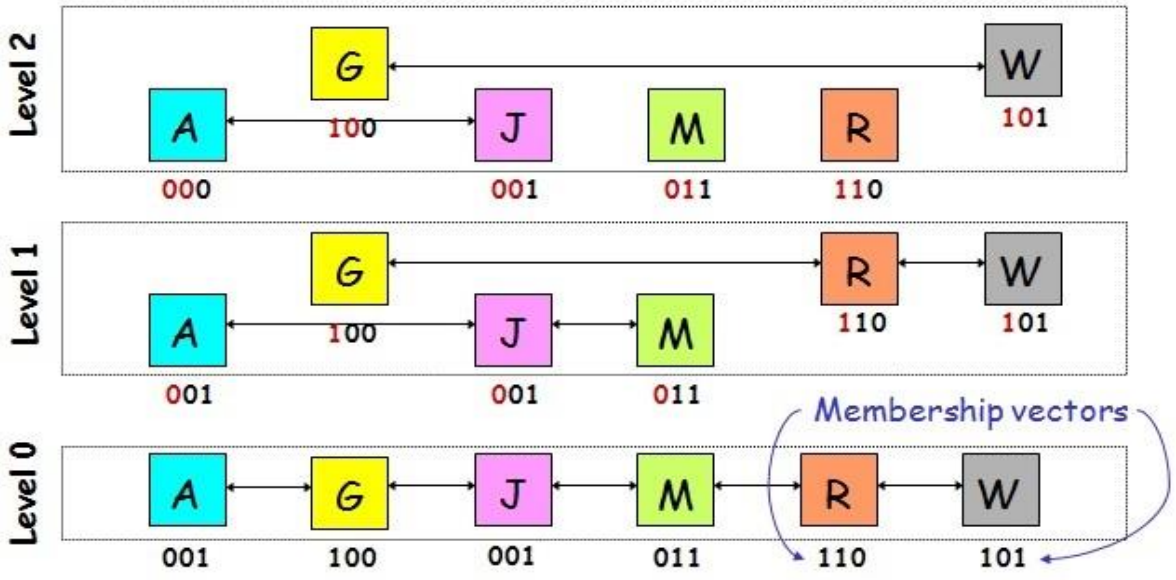

Figure 4.10: Skip Graph for Search algorithm 'M' [30]

To search Node 'M' following steps should be followed:

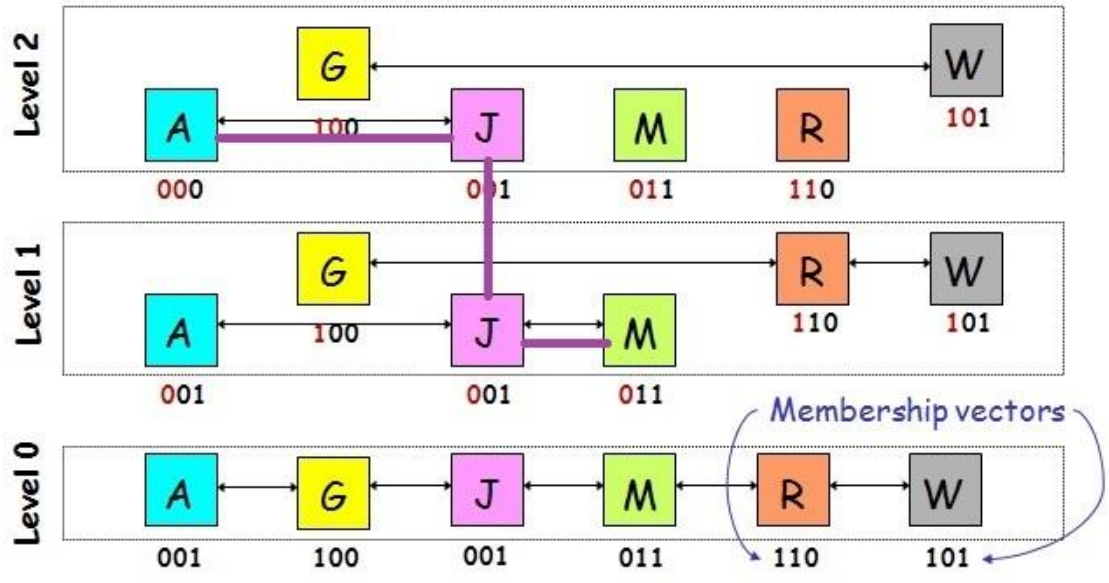

Figure 4.11: Search operation for node 'M'

4.4.3 Algorithm for deletion of a node

To delete a node in a Skip Graph:

Input:

- Key value of node as “key”

Output:

- Node removed from the Skip graph and all links updated

Deletion of Node From Skip Graph

When node ‘u’ wants to leave the network, it deletes itself in parallel from all lists above level 0 and then deletes itself from level 0.

Algorithm 3: delete for existing node u

```
1 for ℓ ← 1 to u.maxLevel in parallel do
2 |   delete u from list at level ℓ
3 delete u from list at level 0
```

Figure 4.12: Deletion algorithm [4]

Example of deletion of node from Skip Graph

First the deletion node should be identified on all level . Suppose Node ‘J’ needs to be deleted from the graph .

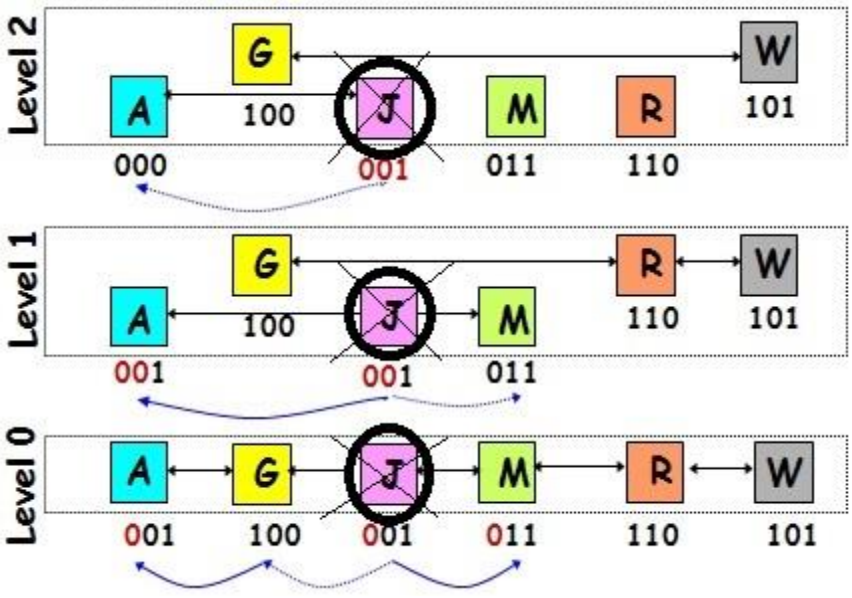

Figure 4.13: Deletion operation in skip Graph

- Start deletion process from Level 1 to maxLevel . Delete selected node from all these levels.
- At last delete node from Level 0 .

Once node ‘J’ is deleted from the Skip Graph updation are done in the links and membership vectors.

Skip graph obtained after deleting Node ‘J’ is (Figure 4.14):

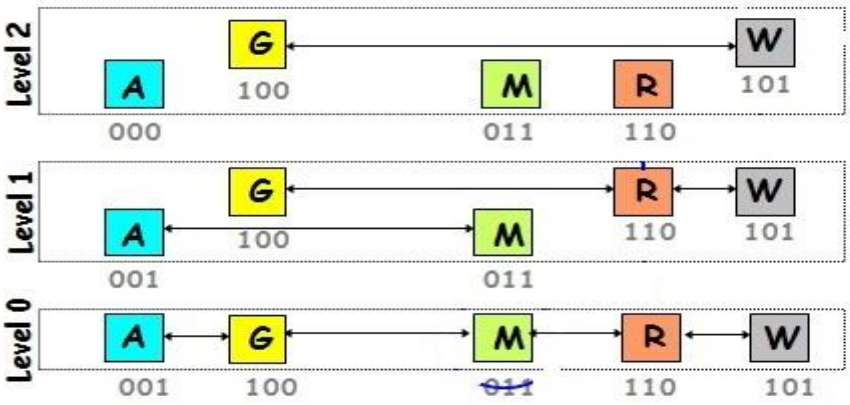

Figure 4.14: Graph after deleting node ‘J’

4.4.4 Complexity Analysis of Algorithms

In Skip Graph all nodes are connected using doubly Linked list. If operations on doubly-linked list can be treated as atomic, and that it supports search, insert, and delete operations that each take O(k) time from a node k hops away from its target. So in this thesis one of very important assumption is that operations on different doubly-linked lists can be performed in parallel without interference. So there is no overhead of doubly link list operation in algorithms.

In Insertion operation in Skip Graph with n nodes , it takes O(log n) time and O(logn) messages transfer. The search operation in a skip graph with n nodes takes expected O(log n) messages and O(logn) time. In the absence of concurrency, the delete operation in a skip graph S with n nodes takes expected O(log n) messages and O(1) time[4].

4.5 Factors related to nodes participating in Proximity and Centrality Analysis

Factors which would decide the proximity of a node to all its neighbors depends on the being application used ,for example, if there are friendship networks then analysis would depend upon following factors:

- No. of visits to a particular person's profile
- No. of likes or comments for each node
- Directly linked or not (no. of nodes between two node)
- Active daily hours on Website
- Pages which are regularly visited
- Friend requests received or sent
- Applications or games used or created

The factors that play important role in analysis of the network can be increased, decreased modified depending on the analysis's requirement or type of analysis. So the significance of Result changes with the change in selection of parameters for analysis.

4.6 Software Used for Automatic Network Generation

Pajek

PAJEK is a drawing package for Windows for analysis and visualization of large networks having some thousands or even millions of vertices. It was jointly developed in 1996 by Vladimir Batagelj and Andrej Mrvar from university of Ljubljana, Slovenia. PAJEK is implemented using Delphi program and runs on Windows operating system. It is freely available for non-commercial use and can be downloaded free of charge from its home pages at [22].

In Slovenian language the word Pajek means spider. The main motivation for development of Pajek was the observation that there exist several sources of large networks that are already in machine-readable form. Pajek provides tools for analysis

and visualization of such networks: collaboration networks, organic molecule in chemistry, protein-receptor interaction networks, genealogies, Internet networks, citation networks, diffusion (AIDS, news, innovations) networks, data-mining (2-mode networks), etc.

The main goals in the design of **Pajek** are:

- To support abstraction by (recursive) decomposition of a large network into several smaller networks that can be treated further using more sophisticated methods.
- To provide the user with some powerful visualization tools.
- To implement a selection of efficient (subquadratic) algorithms for analysis of large networks.

Besides ordinary (directed, undirected, mixed) networks Pajek supports also multi-relational networks, 2-mode networks (bipartite (valued) graphs – networks between two disjoint sets of vertices), and temporal networks (dynamic graphs networks changing over time).

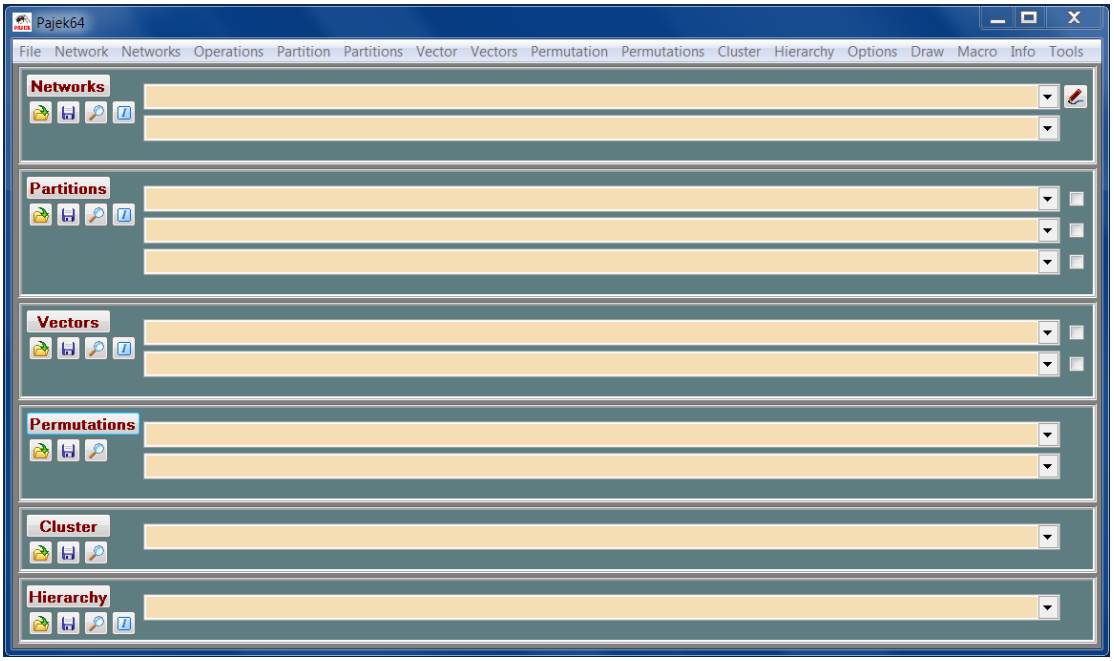

Figure 4.15: Main window of Pajek

PAJEK visualization and analysis of large networks are performed using six data type [22] shown in Figure 4.15 :

- **Network (graphs)**-the main object (vertices and lines). Default extension .net.
- **Partition**- (nominal or ordinal properties of vertices and ordinal properties). Default extension .per.
- P**ermutation** (re-ordering of vertices and ordinal properties). Default extension .per.
- **Cluster**-(subset of vertices e.g. a class from partition). Default extension .cls.
- **Hierarchies** (general tree structure on vertices)-hierarchically ordered clusters and vertices. Default extension .hie.
- **Vector** (numerical properties)- values of vertices. Default extension .vec.

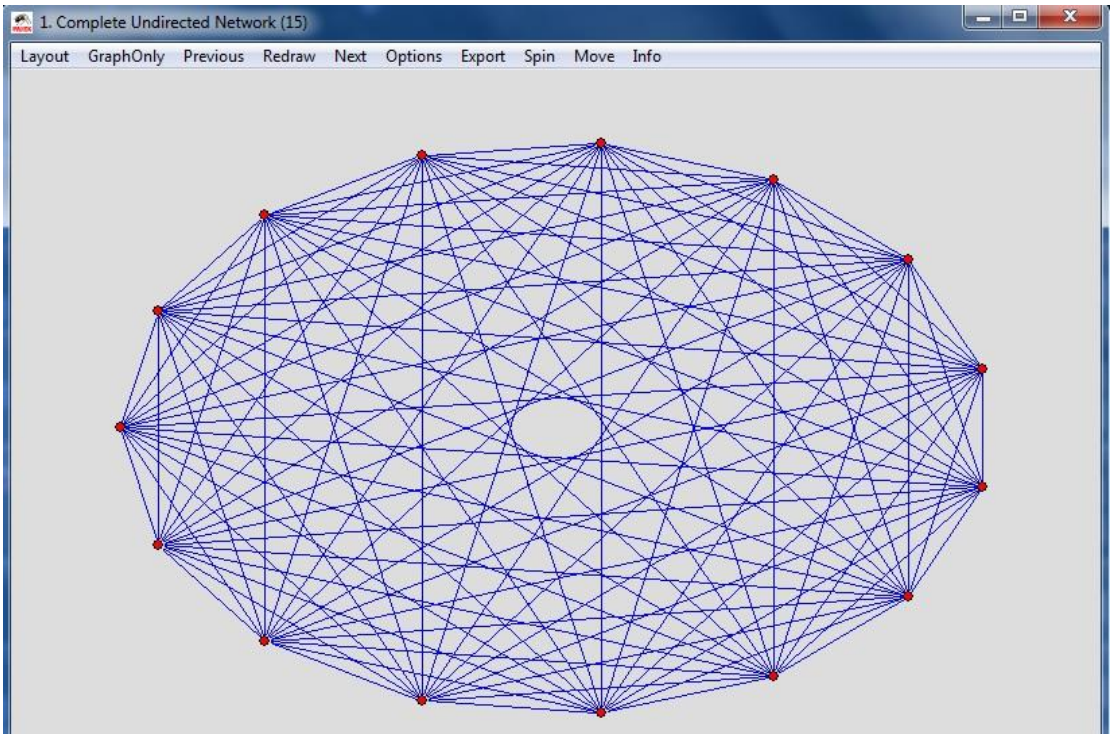

Figure 4.16: Network Created by Pajek

PAJEK strengths include the ability to:

- Contain several transformations that support different transitions among these data structures. The menu structure of the main PAJEK‟s window is based on them.
- Use its main window as a calculator paradigm with list-accumulator for each data type. The operations are performed on the currently active (selected) data and are also returned with the results through accumulators.
- Create special emphasis to automate generation of network layouts. Many standard algorithms for automatic graph drawing are implemented such as spring embedders, layouts by eigenvalues/eigenvectors drawing in layers (genealogies and other acyclic structures), fish-eye view and block (matrix) representation. Properties of vertices and edges (given as data or computed) can be represented using colors, sizes and shapes of vertices and edges.
- Support drawing sequence of networks in its Draw window and exports sequences of network in suitable out graphic formats (Bitmap, Encapsulated PostScrpt (EPS), Scalable output vector Graphics (SVG), Virtual Reality (VRML), MDL MOL/Chime and Kinemages) that can be examined with special 2D or 3D viewers.

Chapter 5

Implementation and Results

5.1 Generation of network using Pajek

Pajek64, a network analysis tool, used to create, draw and analyze different types of networks is used for automatic network generation.

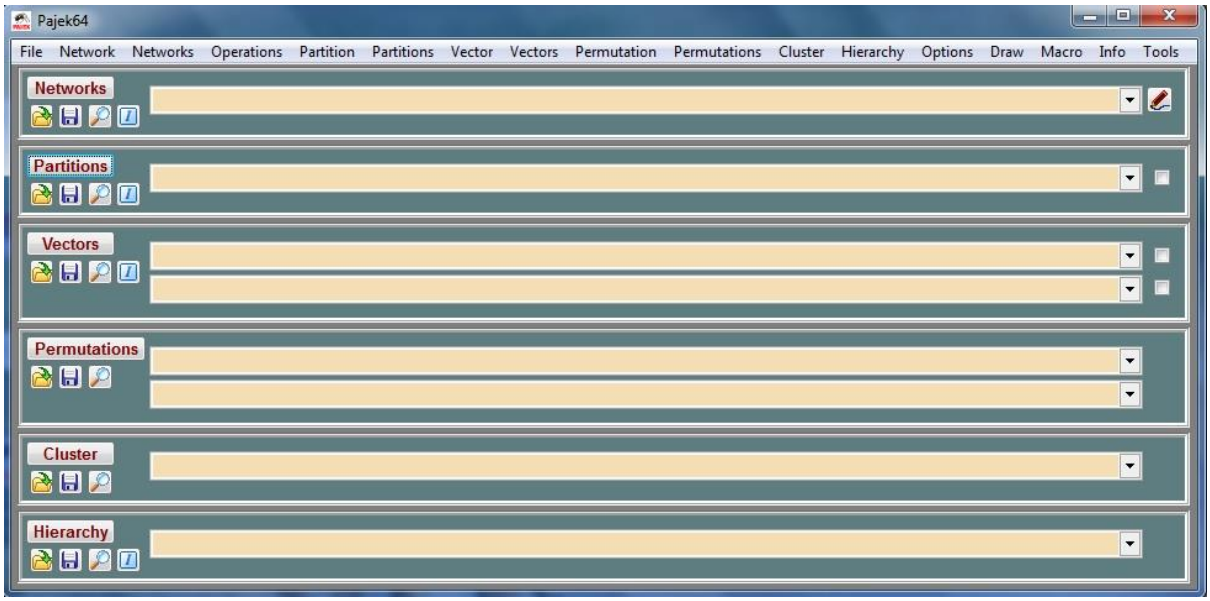

Figure 5.1: Main Window of Pajek

Pajek has various options to draw a network and a complete undirected network of 10 nodes is drawn.

Network -> Create New Network-> Complete Network->Undirected

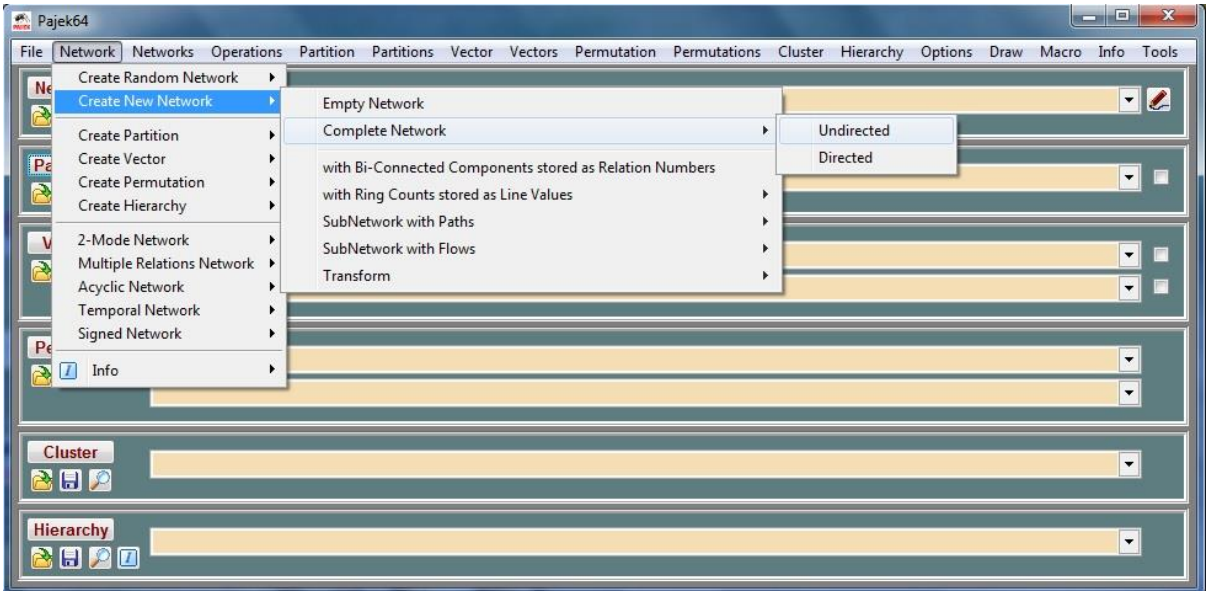

Figure 5.2: Steps to Create Network in Pajek

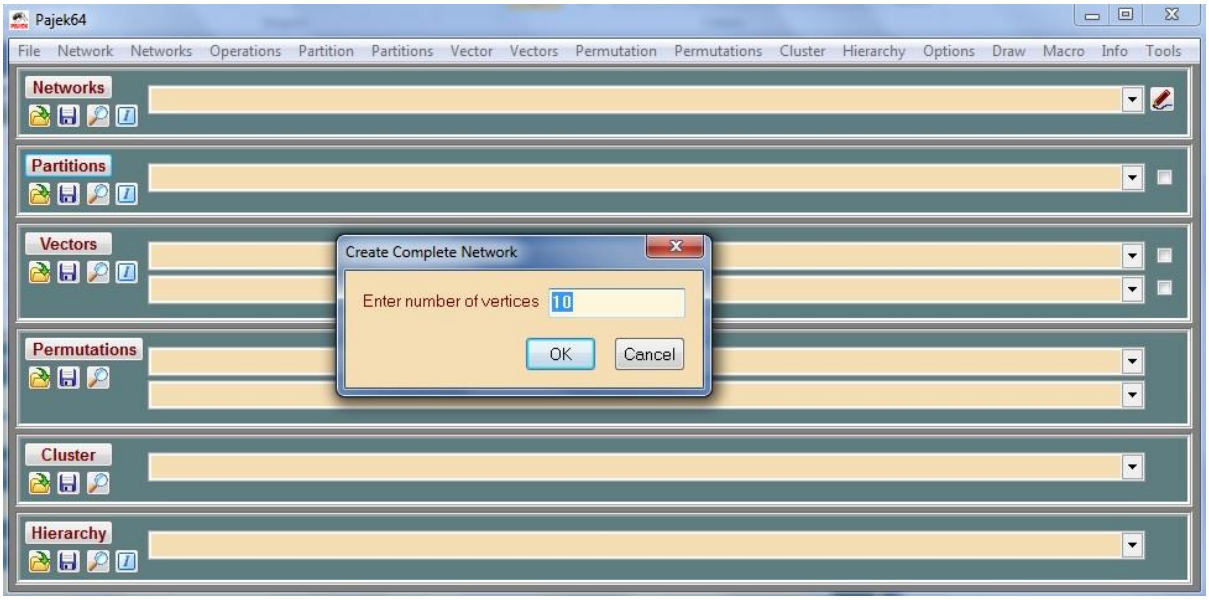

Figure 5.3: No. of vertices selection for Network

Figure 5.3 represent a network of 10 nodes considered for experimentation and implementation.

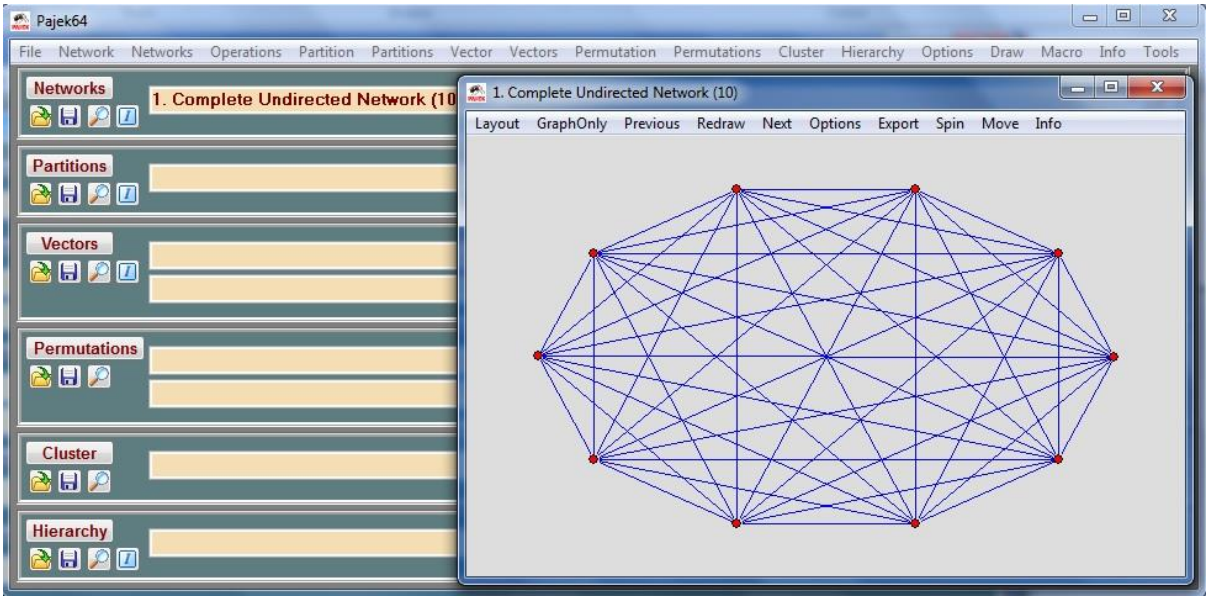

Figure 5.4: 10 Node Complete Undirected Network in Pajek

The factors that influence analysis of the network can be increased, decreased modified depending on the analysis's requirement. Value of each factor would be taken from data base in real time, but for experimental purposes, the values for parameters have been assigned randomly and three parameters have been considered in the friendship network:

i. No. of like to a particular node (*e.g.* No. of like given by node 1 to node (2,3,4..)

ii. No. of visits to particular persons profile (*e.g.* No. of time node 2 is visited by node 1)

iii. No. of nodes between any 2 nodes (Distance)

Various modes available for viewing the network generated in Figure X are:

1-- for Binary Matrix

2-- for Adjacency Matrix

3-- for Adjacency List

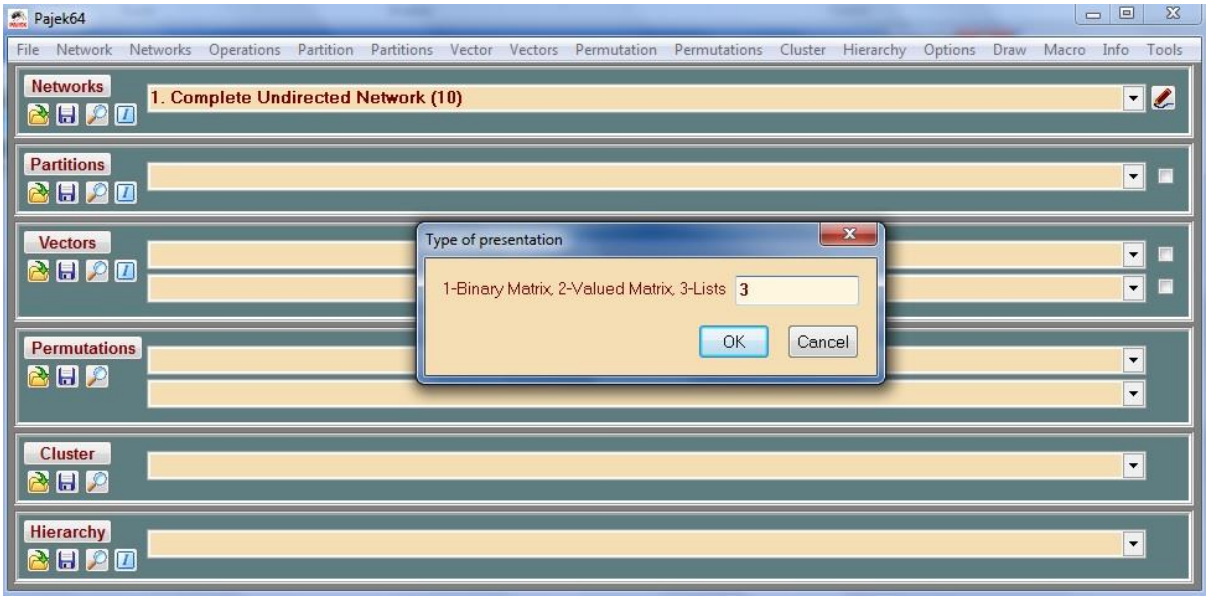

Figure 5.5: Select type of representation of Network File

Graph in Adjacency List option.

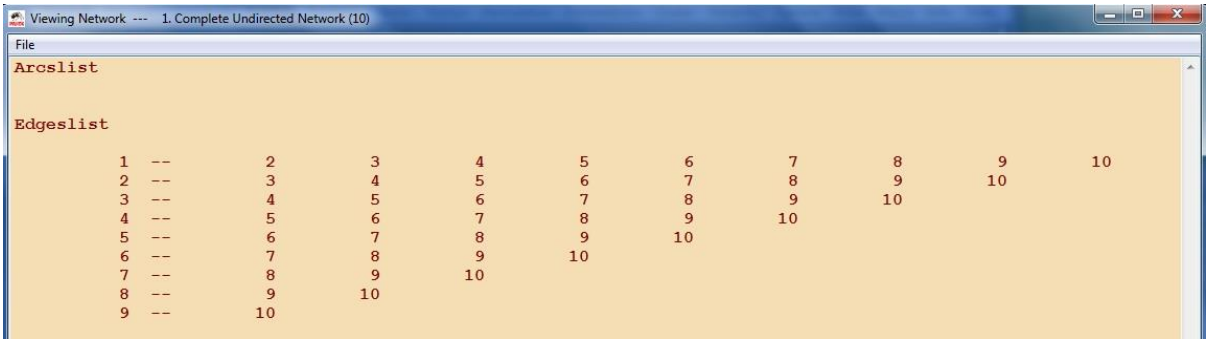

Figure 5.6: Adjacency List Representation

Graph in Adjacency Matrix option.

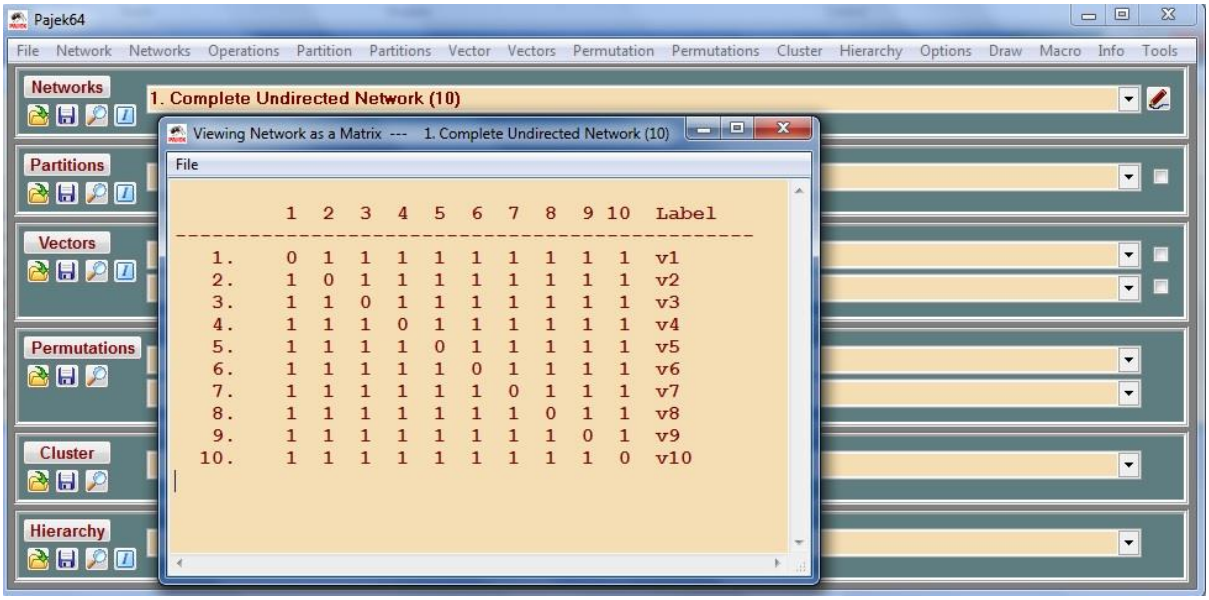

Figure 5.7: Adjacency Matrix Representation

For Skip Graph program we take Adjacency List file as input file.

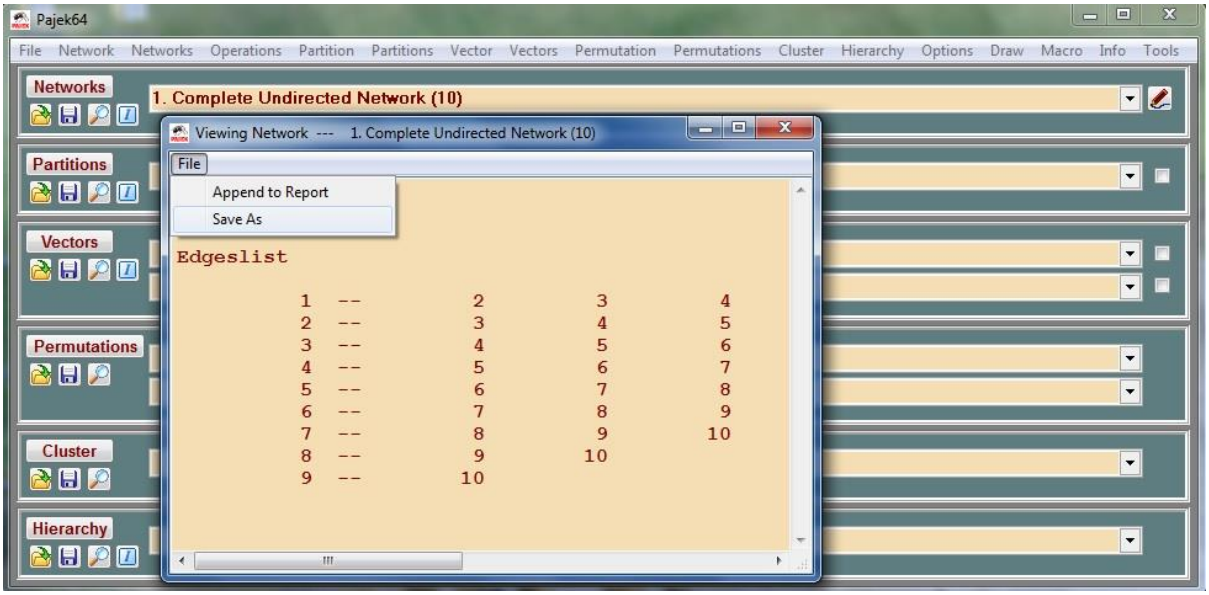

Figure 5.8: Steps to Save Adjacency List File

5.2 Creation of Skip Graph

This "p1.rep" File in our program is network adjacency list file from Pajek, that act as input file in our program.

```
int main()
{    int a[100]={0};
     int size = 6;
     int start =1;
     FILE *f;

     f = fopen("p1.rep","r");
```

Figure 5.9: Skip graph creation for related input .

As Skip Graph is a probability based data structure, every time the program is run we get different arrangement of nodes at different levels and different no. of levels.

Pattern 1:

```
asingh@ubuntu:~/Desktop/skipgraph/skip2$ ./a.out

Displaying skipset

Nodes in the skipgraph are :

[ Level 3 ] <-> 6
[ Level 2 ] <-> 6
[ Level 1 ] <-> 2 <-> 4 <-> 6 <-> 7 <-> 8 <-> 9
[ Level 0 ] <-> 1 <-> 2 <-> 3 <-> 4 <-> 5 <-> 6 <-> 7 <-> 8 <-> 9 <-> 10
```

Figure 5.10: Skip Graph Creation Pattern 1

- Pattern 2:

```
asingh@ubuntu:~/Desktop/skipgraph/skip2$ ./a.out

Displaying skipset

Nodes in the skipgraph are :

[ Level 6 ] <-> 9
[ Level 5 ] <-> 5 <-> 9
[ Level 4 ] <-> 5 <-> 9 <-> 10
[ Level 3 ] <-> 5 <-> 9 <-> 10
[ Level 2 ] <-> 2 <-> 5 <-> 8 <-> 9 <-> 10
[ Level 1 ] <-> 2 <-> 3 <-> 5 <-> 6 <-> 8 <-> 9 <-> 10
[ Level 0 ] <-> 1 <-> 2 <-> 3 <-> 4 <-> 5 <-> 6 <-> 7 <-> 8 <-> 9 <-> 10
```

Figure 5.11: Skip Graph Creation Pattern II

5.3 Proximity and Centrality Analysis

In our experimental work on SNA, friendship network designed in Pajek is considered using above mentioned three parameters and values are assigned randomly.

To assign values randomly following algorithm is used:

```
Algorithm to Assign Value to the parameters
INPUT - No. of parameters
OUTPUT  - Assign random values to all parameters with "random" function
for i -> 1 to size do
      for j -> 1 to size do
        if( i !=j) then
          like[i][j] = random();
          visit[i][j] = random();
          distance[i][j] = random() ;
          sum.like[i] = sum.like[i]+like[i][j]
          sum.visit[i] = sum.visit[i]+visit[i][j]
      end if
   end for //j
end for//i
```

Figure 5.12: Algorithm for random value assignment

Some factors in analysis are directly related or some factors are inversely related to the analysis, so formula of calculation of analysis is depend upon the selection of parameters . These parameter act as base for further analysis part.

```
Algorithm to calculate Proximity Analysis
INPUT - value of all parameters for node
OUTPUT - Value of proximity for each node
for i -> 1 to size do
        for j -> 1 to size do
          if( i !=j) then
            p = (((sum.like[i] / like[i][j]) * (sum.visit[i] / visit[i][j])) / distance[i][j])
             print proximity of node i with j is p
              end if
   end for //j
end for// i
```

Figure 5.13: Algorithm for proximity Analysis

```
Algorithm For Centrality analysis
INPUT - Data from proximity analysis
OUTPUT - Find Most famous node in the graph
for i -->i to size do
    c[i] = c[i] +p[i] // calculate total proximity of one node with all others
end for
find the highest value from array c[i]
print i is result of Centrality analysis
```

Figure 5.14: Algorithm for Centrality Analysis

Significance of the analysis may change as per the selection of parameters and their interpretation in formulas. For example one can use this analysis to find most famous person in a friend circle, or to check any suspect in criminal activity , or find the source of disease spreads in a particular area etc .

Results of Proximity Analysis

Proximity of each node with its neighboring node is represented in figure 5.15.

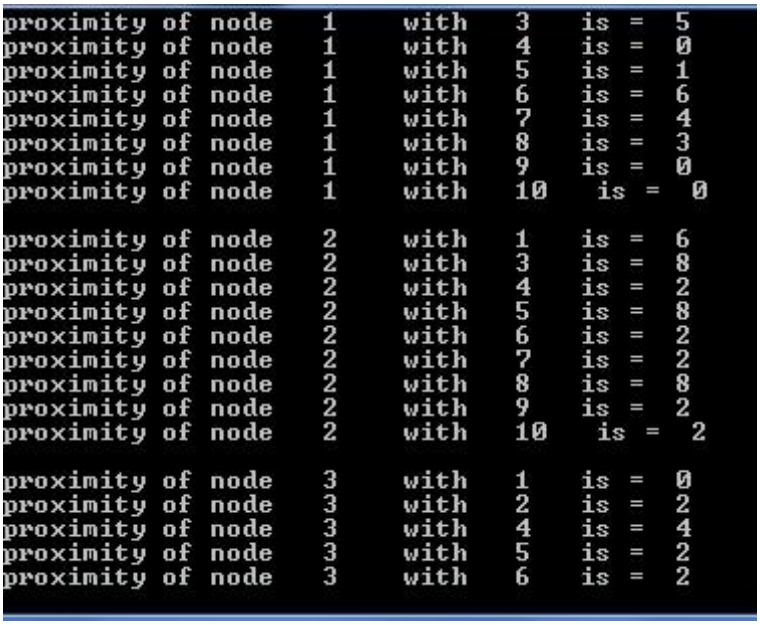

Figure 5.15: Results of proximity Analysis

Results of Centrality analysis

To find the central node of the entire network, results achieved from figure 5.16 are analysed and node having highest overall proximity is registered as head of the entire network.

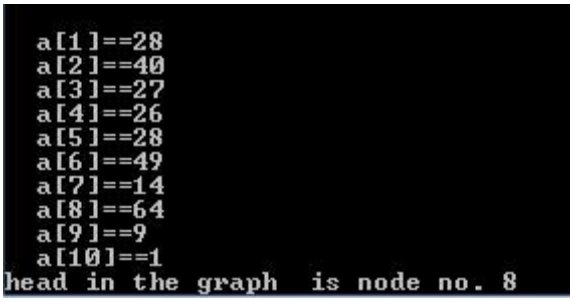

Figure 5.16: Results of Centrality Analysis

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

Social networks are represented as graphs where each node called an actor or vertex in a graph represents an individual person or a group of persons and the fundamental component of SNA is the relationship defined by these linkages among units or nodes in the network. An understanding of the structure of online social networks can benefit the design of new systems and helps in understanding the impact of online social networks on the future Internet.

The major focus of the thesis has been on two issues: first to identify a data structure for the efficient storage and retrieval of graphs and second was proximity analysis. The first issue has been resolved by using Skip Graph. It has been experimentally evaluated that the space used for storing a graph is tremendously decreased by using skip graphs as they use linked list and pointers. An algorithm has been designed to identifying the head or the lead node of a group based on some selected parameters. Further, the factors that play important role in analysis of the network can be increased, decreased modified depending on the analysis's requirement. Such analysis can be helpful in identifying trends, likings and disliking of a particular group in the large communities. In our work the test bed for all experimental set up was friendship network, but this can be easily extended to other networks such as transport network, railway network, chemical compounds, etc.

6.2 Future Scope

Our work was limited to few nodes but when we talk about social networks it covers million of nodes. The data structure used and the proximity analysis done can be extended to large number of parameters and for more scalability up to million and billions of nodes. It can be further extended in real time environment in which we dynamically fetch data and perform analysis. Secondly the storage schema can be upgraded with any new storage technique where it would require lesser storage space as well as lesser access time leading to further optimization of graph storage and

cluster storage. Another important extension to this work can be in the research areas like link prediction and real time dynamic social network analysis.

References

[1] L. Arge, D. Eppstein, and M.T. Goodrich. "Skip-webs: efficient distributed data structures for multi-dimensional data sets. " , *In Proceedings of the annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pp. 69–76, 2009.

[2] J. Aspnes and G. Shah. "Skip graphs. ", *In Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pp 384– 393, 2003.

[3] J. Aspnes and G. Shah. "Skip graphs. ", *ACM Transactions on Algorithms*, 3(4):37, November 2007.

[4] James Aspnes and Udi Wieder, " The expansion and mixing time of skip graphs with applications. ", In *SPAA '05:In Proceedings of the seventeenth annual ACM symposium on Parallelism in algorithms and architectures*, New York, NY, USA, ACM , pp 126–134, 2005.

[5] Marin, Alexandra and Barry Wellman. Forthcoming. "Social Network Analysis: An Introduction." , in Peter Carrington and John Scott (eds.). Handbook of Social Network Analysis. London Sages, 2010

[6] K. Tsuda, H. Saigo."Graph Classification, in Managing and Mining Graph Data", Springer , 2010.

[7] A. W. Lim, S.-W Liao, and M. S. Lam. "Blocking and array contraction across arbitrarily nested loops using affine partitioning", In Proceedings of the eighth ACM SIGPLAN symposium on Principles and practices of parallel programming, ACM Press, pp. 103-112, 2001.

[8] W. Thies, F. Vivien, J. Sheldon, and S. Amarasinghe. "A unified framework for schedule and storage optimization", In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation, ACM Press, pp. 232-242, 2001.

[9] A. Cohen, V. Lefebvre, "Optimization of storage mappings for graphs", Technical Report, PRiSM, U. of Versailles, pp 40-46,1998.

[10] G. Pike. "Reordering and Storage Optimizations for graphs", PhD thesis, University of California, Berkeley, 2002.

[11] A. Cohen. "Parallelization via constrained storage mapping optimization", ACM,1999.

[12] "Simple Network example" [online]. Available: www.google.com [Accessed 28 March 2013].

[13] "Data Structure to Store graph wiki" [online]. Available: https://en.wikipedia.org/wiki/Adjacency_matrix [Accessed 30 March 2013] .

[14] A. A. Nanavati, S. Gurumurthy, G. Das, D. Chakraborty, K. Dasgupta, S. Mukherjea, A. Joshi. “On the structural properties of massive telecom call graphs: findings and implications”, In CIKM ’06: Proceedings of the 15th ACM 45 international conference on Information and knowledge management, New York, NY, USA,ACM , pp 435-444, 2006.

[15] M. Akiko, R. Rudy. “Classifying positions in online debates from reply activities and opinion expressions”, In Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China, pp. 869–875, 2010.

[16] M. Rattigan, M.Maier, D. Jensen. “Graph Clustering with Network Structure”, ICML, 2007.

[17] Eunice E. Santos, Chair, Elisa D. Sotelino, Yang Cao, Ezra Brown , “Effective and Efficient Methodologies for Social Networks”, Thesis Submitted In Ph.d , Blacksburg, USA ,Dec , 2007

[18] L. Euler. “Solutio problematis and geometriam situs pertinentis”, published in *Commentarii academiae scientiarum Petropolitanae* 8, pp. 128-140, 1741.

[19] B. Bollobas.“Modern Graph Theory”, Springer Verlag, New York, USA. 1998.

[20] S. Wasserman, K. Faust. “Social Network Analysis”, Cambridge University Press, Cambridge, UK. 1994

[21] J. Scott. “Social Network Analysis: A Handbook”, SAGE Publications, London, UK. 2000

[22] Andrej Mrvar and Vladimir Batagelj Ljubljana, “Programs for Analysis and Visualization of Very Large Networks”, Reference Manual April 2, 2013

[23] L. S. Lasdon. “Optimization Theory for Large Systems”, MacMillan, New York. 1970

[24] T. P. Shabeera, Priya Chandran, S D Madhu Kumar , “Authenticated and Persistent Skip Graph: A Data Structure for Cloud Based Data-Centric Applications. ”, Chennai, India , ACM , 2012

[25] H. Mendes , C.G. Fernandes , “A Concurrent Implementation of Skip graphs. ”, Electronic Notes in Discrete Mathematics 35 pp .-263-268,2009.

[26] J. Aspnes , U. Wieder , “The expansion and mixing time of skip graphs with applications. ”, Springer-Verlag , pp 385-394 , 2008.

[27] Thomas Clouser, Mikhail Nesterenko, Christian Scheideler , “Tiara: A self-stabilizing deterministic skip list and skip graph. ” , Elsevier , 2012.

[28] Fuminori Makikawa, Tatsuhiro Tsuchiya, Tohru Kikuno , “Balance and Proximity-Aware Skip Graph Construction. ” , First International Conference on Networking and Computing, 2010.

[29] Jianjun Yu, Hao Su, Gang Zhou, Ke Xu , “SNet: Skip Graph based Semantic Web Services Discovery .” Seoul, Korea. ACM , 2007.

[30] James Aspnes , Guari Shah, ppt on “ Skip Graphs ” in SODA Available: http://www.cs.yale.edu/homes/aspnes/papers/skip-graphs-soda03.ppt, 2003.

[31] Michael T. Goodrich, Michael J. Nelson , Jonathan Z. Sun , “The Rainbow Skip Graph: A Fault-Tolerant Constant-Degree P2P Relay Structure. ”, ArXiv - 2009

[32] J. Ian Munro and Patricio V. Poblete, “Fault tolerance and storage reduction in binary search trees. ”, *Information and Control*, 62(2/3):210-218, August 1984.

List of Publications

Amritpal Singh, Dr. Shalini Batra, "A short survey of Advantages and Applications of Skip graphs ", in the INTERNATIONAL JOURNAL OF MANAGEMENT & INFORMATION TECHNOLOGY, July Edition, (Communicated).