December 2016

Identification of key players in networks using multi-objective optimization and its applications

R Chulaka Gunasekara

Syracuse University

Abstract

Identification of a set of *key players*, is of interest in many disciplines such as sociology, politics, finance, economics, etc. Although many algorithms have been proposed to identify a set of key players, each emphasizes a single objective of interest. Consequently, the prevailing deficiency of each of these methods is that, they perform well only when we consider their objective of interest as the only characteristic that the set of key players should have. But in complicated real life applications, we need a set of key players which can perform well with respect to multiple objectives of interest.

In this dissertation, a new perspective for key player identification is proposed, based on optimizing multiple objectives of interest. The proposed approach is useful in identifying both key nodes and key edges in networks. Experimental results show that the sets of key players which optimize multiple objectives perform better than the key players identified using existing algorithms, in multiple applications such as *eventual influence limitation* problem, *immunization* problem, improving the fault tolerance of the *smart grid*, etc.

We utilize multi-objective optimization algorithms to optimize a set of objectives for a particular application. A large number of solutions are obtained when the number of objectives is high and the objectives are uncorrelated. But decision-makers usually require one or two solutions for their applications. In addition, the computational time required for multi-objective optimization increases with the number of objectives. A novel approach to obtain a subset of the Pareto optimal solutions is proposed and shown to alleviate the aforementioned problems.

As the size and the complexity of the networks increase, so does the computational effort needed to compute the network analysis measures. We show that *degree centrality* based network sampling can be used to reduce the running times without compromising the quality of key nodes obtained.

IDENTIFICATION OF KEY PLAYERS IN NETWORKS USING MULTI-OBJECTIVE OPTIMIZATION AND ITS APPLICATIONS

By

Raigamage Chulaka Gunasekara

B.Sc (Hons.) in Computer Science and Engineering, University of Moratuwa, 2009
M.Sc. in Computer Science, Syracuse University, 2015

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer & Information Science & Engineering

Syracuse University
December 2016

ACKNOWLEDGMENTS

There is a long list of wonderful individuals to whom I should be extremely thankful for helping me throughout the journey towards completing this thesis.

First and foremost, I am extremely fortunate to have two impeccable advisors; Professor Kishan Mehrotra and Professor Chilukuri K. Mohan, who have been two pillars of driving force, encouragement, and support throughout this journey. I thank my two advisors for all the fruitful research discussions, the time they were able to fit into their busy schedules when I needed, and helping to develop this study into a PhD dissertation. I am also grateful to my dissertation committee members; Professor Vir Phoha, Professor Sucheta Soundarajan, Professor Edmund Yu and Professor Utpal Roy for their time and effort in providing me with invaluable feedback in putting together and improving my dissertation.

I should also acknowledge the former and current lab mates of the SENSE lab at Syracuse University for very fruitful weekly research meetings, their valuable feedback on my work, and for all the good times spent in Syracuse. I should also thank all my friends in Syracuse, for being a integral part of my life for the last five years.

I am also thankful for all the Professors at the Department of EECS at Syracuse University from whom I have learned invaluable new knowledge and numerous skills. I should also thank the staff members at the Department of EECS for seamlessly handling all administrative work necessary throughout my stay at Syracuse University. I consider myself very fortunate to have had more some fantastic teachers right from my early school days in Sri Lanka. I take this as an opportunity to thank all my teachers at Ananda College, Colombo and all my lecturers at University of Moratuwa, Sri Lanka.

Most of all, I am grateful to my family. My wife Ishani Ratnayake; who has been selfless in supporting me every step of the way and waiting up many late nights throughout last five years. I am extremely thankful to my parents and parents in law, who have been the source of encouragement and influence throughout my life. If not for their love, support, and encouragement, none of this could have happened. I am dedicating this thesis to my precious daughter Mithuli, for being the source of unending joy and love of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Networks provide an excellent platform to model many complex systems comprised of a set of entities and various relationships among them. When such a system is modeled as a network, the entities are represented as nodes (vertices), and the relationships among the entities are represented as edges. Some such systems which are commonly modeled and studied as networks include the Internet, social networks, food web, gene regulatory networks, infrastructure networks, etc. Various concepts of network theory have been used to reveal interesting patterns and open pathways to insights in the modeled systems. Starting from the well known 'Seven Bridges of Koenigsberg' problem in 1735, network theory has been applied to many disciplines including physics, computer science, electrical engineering, biology, economics, and sociology.

Some of the interesting problems that are studied through network science include identification of important entities in systems, detecting communities of users in social networks, identifying anomalous users in social settings, modeling and analyzing information or disease spread among people in various regions, and predicting possible connections that can occur in future among people.

Modeling real-world data using networks has become popular in recent years, and the size of the networks analyzed has also grown rapidly in size. For example, social networks such as Facebook and Twitter have reached hundreds of millions of users around the world. The World Wide Web contains at least five billion pages. As the networks grow in size, the insights that can be unveiled through network science expand, while presenting the researchers with challenges that arise with such large volumes of data.

1.1 Objectives

The main focus of this dissertation is to *identify the most important entities in a system modeled as a network.* These important entities are referred to as *key players*, throughout this dissertation. Networks consist of *nodes* and *edges*; in this study we consider identification of both the *key nodes* and the *key edges*.

Key nodes in an environment represented by a network are the most important entities in the modeled environment; such as decision makers in an organization, opinion leaders in social media, celebrities and political leaders, key infrastructure nodes in an urban network, and mediators between communities.

Selecting a set of key nodes from a system that is represented as a network is an important research problem in many disciplines, such as the following:

- In viral marketing, it is important to identify and target the ‘right’ set of key people in a population to spread information efficiently and effectively.
- In human resource management, it is critical to identify and strategically place the key people to improve the productivity of the entire organization.
- In politics, it is necessary to gain the support of key individuals to gain advantages in political campaigns.

Many *node centrality* measures have been proposed to capture the different behaviors a node can have in a social setting. These node centrality measures are used to identify key nodes in a networks, and are discussed in Chapter 2.

Identifying the important edges in a network plays an important role in applications such as the following:

- In information diffusion applications, it is important to identify the edges which play an important role in diffusing information more efficiently among different parts *(communities)* in the network.
- In determining and strengthening robustness of networked environments, it is important to identify the connections that upon removal would collapse the network, and take necessary measures to protect these connections from attacks and failure.

Current approaches proposed to identify key edges include methods based on the strength of the edges, *edge centrality* measures and optimization techniques addressing a specific property of interest to the network. These measures are discussed in Chapter 2 and Chapter 6.

One common property of all the current approaches for key player identification is that each of these methods identifies key players based on a single characteristic of interest. For example, one trivial method to identify important nodes in a network is to count the number of edges incident on each node. Intuitively, an important node in a network should be connected to more peers in a network setting. But, this method of key node identification does not consider the structure of the network and the positions to which these selected nodes belong in the network. Hence, the identified key players might come from the periphery of the network, which may not correspond to the most important positions in a network.

We investigate the effects of using key players which optimize multiple properties of interest in different well known applications of network science. Our hypothesis is that when a set of key players optimizes multiple properties which are relevant for a particular application, this set of key players should outperform the sets of key players identified based on a single property of interest. Towards this goal, we identify both *key nodes* and *key edges*, that optimize multiple properties of interest. In multiple applications, we show that key players which optimize multiple objectives perform better than the key players identified using existing algorithms.

We utilize *multi-objective optimization* algorithms (such as *NSGA-II*) to optimize a set of objectives for a particular application. Such an approach identifies a set of solutions (rather than one solution needed by a typical decision-maker), which are 'equally' good in optimizing the set of objectives. In addition, the computational time required for multi-objective optimization increases with the number of objectives. To alleviate these problems, we propose a technique which approximates the solutions in the multi-objective optimization. The proposed approach utilizes a two-step process to multi-objective optimization, and has advantages such as: (1) reducing the number of solutions in the solution space, (2) reducing the computational time significantly, and (3) providing solutions which deliver performance 'similar' to the performance obtained by the solutions given by the unmodified multi-objective optimization algorithm.

As the size and the complexity of the networks increase, so do the computational times associated with the network analysis measures. Hence, we focus on how network sampling can be used to reduce the running times without compromising much on the quality of key nodes obtained. We introduce the idea of *degree centrality* based sampling to reduce the running time of the key node identification problem. We show that the multi-objective key player sets obtained with degree centrality based sampled networks perform better than single objective key player sets identified by applying the algorithms on the entire network.

1.2 Organization of the Dissertation

This dissertation is organized as follows. In Chapter 2, we discuss the related work on key node and key edge identification methods. We discuss the widely used centrality measures, and optimization techniques proposed to identify key players in networks.

Following this, in Chapter 3, first we discuss the basics of *evolutionary algorithms*, the background of multi-objective optimization tasks, and discuss how evolutionary algorithms have been used to solve multi-objective optimization. One prevailing issue with multi-objective optimization is that it identifies a set of solutions, rather than a single solution as required by most decision makers. In addition to that, the time complexity of multi-objective optimization increases with the number of objectives. To alleviate these issues, we propose the *leave-k-out* approach for multi-objective optimization. We show that the solutions for multi-objective optimization obtained using our approach perform well compared to other approaches proposed to select solutions from a set of solutions obtained by multi-objective optimization, while reducing the computational time significantly.

In Chapter 4, we propose the algorithm for identifying key nodes which optimize multiple objectives of interest. In our approach we transform the network of interest into a bit string, and apply *leave-k-out* approach for multi-objective optimization to obtain key nodes which optimize multiple properties of interest. We show that by using this approach we can alleviate some of the prevailing problems of key node identification. Then we compare the different key node identification methods in two well known applications, viz., *Eventual Information Limitation (EIL)* and improving the fault tolerance of the *smart grid*, and show that the multi-objective approach outperforms the previous key node identification methods.

As the size and the complexity of the networks increase, so do the running times of the network analysis measures. The proposed multi-objective approach to key player identification depends on the computational complexity of individual network centrality measures and on the computational complexity of evolutionary optimization algorithm employed. The focus of Chapter 5 is on how network sampling can be used to reduce the running times of key player identification without compromising much on the quality of key nodes obtained. First, we give an overview of the common network sampling methods. Next, we propose the idea of degree centrality based sampling approach to reduce the running time of the key node identification problem. Finally, the multi-objective key player sets obtained on degree centrality based sampled networks are used to address two well known problems, viz., *Eventual Information Limitation (EIL)* problem and *Immunization* problem. The results suggest that the multi-objective key player sets identified on sampled networks perform better than single objective key player sets identified by applying the algorithms on the entire network.

In Chapter 6 we address key edge identification. When edges are added to a network, the properties of the network change. The amount of change depends on the importance of the set of edges that are added to the network. In this study, we assume that upon addition a set of *key edges* should maximally improve the network robustness. In this chapter we address the following problem : *Given a network and a budget, how should a set of 'key' edges be selected to be added to the network in order to maximally improve the overall robustness of the network.* Towards this goal, first we discuss the network robustness measures that have been proposed and widely used. Then, we analyze the properties of these robustness measures and identify their similarities and dissimilarities using correlation analysis. Then, we use multi-objective optimization and the *leave-k-out* approach to optimize multiple robustness measures of interest to improve the overall robustness of a network. We provide experimental evidence which shows the improvement in multiple robustness measures when the new edges are added using our algorithm.

Finally, Chapter 7 provides the concluding remarks of this study and the future directions of research.

1.3 Contributions

The main contributions of this thesis are as follows.

1. We are the first to propose an approach that identifies a set of key nodes which optimize multiple properties of interest. The experimental results show that the key nodes identified using this approach outperform the key nodes identified using other

approaches in multiple well known applications.

2. A key edge identification method, which optimize multiple properties of interest is proposed. The sets of key edges identified using this approach improves the overall robustness a network, compared to previous approaches to key edge identification.

3. We propose a two-step approximation approach for multi-objective optimization. The solutions obtained for multi-objective optimization using our approach perform 'equally' well compared to other approaches proposed to select solutions from a set of solutions obtained by multi-objective optimization, while reducing the computational time significantly.

4. A sampling approach based on *degree centrality* is proposed. We show that on multiple applications, the multi-objective key player sets identified on sampled networks perform better than the single objective key player sets identified by applying the algorithms on the entire network.

CHAPTER 2
KEY PLAYER IDENTIFICATION IN NETWORKS

As discussed in Chapter 1, key player identification in networks is an important problem. Hence, over the years many algorithms have been proposed to solve this problem. This chapter summarizes previous work on key player identification in networks, considering key node identification as well as key edge identification.

2.1 Key node identification in networks

Given a network $G = (V, E)$ with a set of nodes V and a set of edges $E \subseteq V \times V$, network centrality measures assign a value to the nodes in V based on the structural properties of the network. The score each node gets assigned depends on the property of interest. A network centrality measure is a function that maps a node v in the network $G = (V, E)$ to a real number. Based on the value each node receives from the centrality measure, a rank can be assigned to each node. This rank determines the importance of a node with respect to the structural property on which the centrality measure is based. In this section, some of the most widely used centrality measures are presented.

2.1.1 Degree Centrality

The number of vertices adjacent to a given vertex in a network is the degree of that vertex. *Degree centrality* is defined as the ratio of the number of neighbors of a vertex with the total number of neighbors possible [75].

$$C_{degree}(x) = d(x) \tag{2.1}$$

$$C_{degree\ centrality}(x) = \frac{d(x)}{(n-1)} \tag{2.2}$$

where, $d(x)$ is the number of nodes adjacent to node x, and n is the total number of nodes in the network. For networks with directional edges (directed networks), two variants of node degree are defined. In directed networks, the *In-degree* of a vertex x refers to the number of edges received by x, and the *Out-degree* of a vertex x refers to the number of edges initiated by x. Degree centrality is used to rank vertices in a network based on the number of direct connections of each vertex, where the implication is that the vertices that have more direct connections are more important. Degree centrality is a local measure and does not consider the importance of the vertices to whom each vertex is connected; hence using degree centrality to identify key nodes may not be satisfactory in some cases.

2.1.2 Betweenness Centrality

The degree of a node is not the only measure of the importance of a node in a network. In *Betweenness centrality*, the nodes which have a high probability of occurring in shortest paths of other nodes are considered to be more important [45, 81].

$$C_{betweenness\ centrality}(x) = \sum_{y,z \neq x,\ \sigma_{y,z} \neq 0} \frac{\sigma_{y,z}(x)}{\sigma_{y,z}} \tag{2.3}$$

where, $\sigma_{y,z}$ is the number of shortest paths between nodes y and z, and $\sigma_{y,z}(x)$ is the number of shortest paths between y and z that passes through x. The nodes with high betweenness centrality often act as bridges between different communities in a network. Thus, removal of a node with high betweenness centrality can lead to increase in the geodesic path lengths, and in the extreme case, the network might even get disconnected. In real world networks, this can be important; for example, to prevent the spread of a disease in an epidemic network. A common criticism for shortest-paths based measures is that they do not take into account spread along non-shortest paths. Hence, betweenness measures that relax this assumption by including contributions from essentially all paths between nodes (not just the shortest) have also been proposed [83].

2.1.3 Closeness Centrality

In *closeness centrality*, the nodes with smallest paths to other nodes are considered more important, formally defined as the length of the average shortest path between a vertex x and all other vertices in the network [92].

$$C_{closeness\ centrality}(x) = \left[\frac{\sum_{j=1}^{n} d(x,i)}{(n-1)} \right]^{-1} \tag{2.4}$$

where $d(x,i)$ is the shortest path distance between nodes x and i, and n is the total number of nodes in the network. This can be used to measure how long it will take to spread information from node x to all other nodes, and thus plays an important role in information propagation in networks. For disconnected networks, Harmonic Centrality, which is a variant of closeness centrality, has been defined as follows:

$$C_{harmonic\ centrality}(x) = \sum_{x \neq i} \frac{1}{d(x,i)} \tag{2.5}$$

where $\frac{1}{d(x,i)}$ is taken to be 0 for disconnected node pairs.

2.1.4 Eigenvector Centrality

Eigenvector centrality assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node of interest [13]. Unlike degree centrality, in this case the importance of the neighbors is also taken into account.

The eigenvector centralities of all the nodes in the network (vector x) are defined using the equation,

$$Ax = \lambda_1 x \tag{2.6}$$

where A is the adjacency matrix of the network and λ_1 is the highest eigenvalue of A.

The power iteration method is used to approximate eigenvector centrality. Here, the eigenvector centrality of a vertex is iteratively recomputed as the sum of centralities of its neighbors. To begin the power iteration method, it is assumed that vertex i has eigenvector centrality of $x_i(0)$. Then we gradually improve this estimate by employing a Markov model, and continue until no more improvement is observed. The estimate made at step t is defined as,

$$x_i(t) = \sum_j A_{ij} x_j(t-1) \tag{2.7}$$

i.e.,

$$x(t) = Ax(t-1)$$

and,

$$x(t) = A^t x(0) \tag{2.8}$$

2.1.5 PageRank

PageRank is a link analysis algorithm used by the Google Internet search engine, that assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web [94]. The PageRank of a page is defined recursively and depends on the PageRank metric of all pages that link to it. A page that is linked by many pages with high PageRank receives a high rank. The same concept applies to identifying nodes with high PageRank in a network. PageRank can be thought of as approximately a probability distribution representing the likelihood that a random walk in the network will arrive at any particular node.

$$PR(i) = \frac{1-\beta}{n} + \beta \sum_{(j,i)\in E} \frac{PR(j)}{L(j)} \tag{2.9}$$

where n is the number of nodes in the network, β is the damping factor defined for the network.

2.1.6 Katz Centrality

In Katz centrality a weighted count of all nodes that are connected to a certain node is considered. The weight of a path of length d is computed with attenuation factor β^d, where β is the attenuation constant defined for the application [65].

$$k_i = I_{ij} + \beta \sum_j A_{ij} + \beta^2 \sum_j A_{ij}^2 + \beta^3 \sum_j A_{ij}^3 + ...$$

or, in vector notation,

$$k = (I + \beta A + \beta^2 A^2 + \beta^3 A^3 + ...)e$$

$$k = \sum_{i=0}^{\infty} (\beta^i A^i) e$$

$$k = (1 - \beta A)^{-1} e$$

$$(1 - \beta A)k = e \tag{2.10}$$

where, A is the adjacency matrix and e is a unit vector.

2.1.7 HITS Score

Hyperlink-Induced Topic Search (HITS) (*hubs* and *authorities*) is another algorithm that can be used to rank nodes in a network. In the world wide web, a good hub represents a page that points to many other informative pages, and a good authority represents a page that is linked by many different hubs [69].

The algorithm assigns two scores for each page: its authority value, which estimates the value of the content of the page, and its hub value, which estimates the value of its links to other pages. The authority centrality of a node i (x_i) is proportional to the sum of hub centralities of nodes (y_j) pointing to it, and is defined as

$$x_i = \alpha \sum_j A_{ji} y_j \tag{2.11}$$

The hub centrality of a node is proportional to the sum of authority centralities of nodes it points to, and is defined as

$$y_i = \beta \sum_j A_{ij} x_j \tag{2.12}$$

2.1.8 k-Core Score

k-Core score is another recent approach to identify key nodes in a network. The argument for this approach is that the best spreaders in the network reside in the core of the network [68], and are identified by *k-shell decomposition*. The process assigns an integer index or coreness, k_x to each node, representing its location according to k successive layers (shells) in the network. Small values of k_s define the periphery of the network and the innermost network core corresponds to large k_s.

The process of assigning k_s values for each node is as follows.

i. Start by removing all nodes with degree $k = 1$.

ii. After removing all the nodes with $k = 1$, some nodes may be left with one link, then continue pruning the network iteratively until there is no node left with $k = 1$ in the network.

iii. The removed nodes, along with the corresponding links, form a k shell with index $k_s = 1$.

iv. In a similar fashion, iteratively remove the next shell, $k_s = 2$, and continue removing higher-k shells until all nodes are removed.

2.1.9 Identification of sets of key nodes

In some cases it is necessary to identify a set of key nodes, rather than one important node for the whole network. Some of the examples of such cases involve the following:

1. A network comprised of several communities, so the key nodes should ideally come from different parts of the network.
2. A set of key nodes with multiple capabilities are needed to be identified.

The problem of identifying an optimal set of k players is different from the problem of selecting k individuals that are each individually optimal. Ideally, an algorithm that identifies a set of key nodes should identify k key nodes that can 'collectively' perform well. A few methods have been proposed to find sets of key nodes capable of optimizing some performance criterion.

Group centrality

The concept of centrality has been applied not only to single individuals within a network but also to groups of individuals, for example, measures for degree centrality, closeness and betweenness are defined for a group [39]. Using these measures, a group having high centrality will be the key node set. It must be remarked that group centrality can be used not only to measure how 'central' or important a group is, but also in constructing groups with maximum centrality within an organization.

Combinatorial optimization to identify key node sets

Borgatti [14] described how to find a set of key nodes considering two different aspects. He defined the set of nodes maximally connected to all other nodes as *KPP-Pos* and the set of nodes whose removal would result in a residual network with the least possible cohesion as *KPP-Neg*.

i. KPP-POS - These are the key nodes for the purpose of optimally diffusing something through the network by using the key nodes as seeds. A measure for identifying the measure of reach (D_R), for a set of k key nodes was defined as follows.

$$D_R = \frac{\sum_j \frac{1}{d_K(j)}}{n} \tag{2.13}$$

where, $d_K(j)$, is the minimum distance from any member of set of nodes K to node j, and n is the total number of nodes in the network. The set of k nodes which gives the highest D_R is considered to be the k key nodes for KPP-POS.

ii. KPP-NEG - These are the key nodes for the purpose of disrupting or fragmenting the network by removing the key nodes. A measure of Fragmentation(D_F), for a set of k key nodes was defined as follows.

$$D_F = 1 - \frac{2\sum_{i>j} \frac{1}{d_{ij}}}{n(n-1)} \tag{2.14}$$

where, d_{ij}, is the minimum distance between nodes i and j, and n is the total number of nodes in the network. The set of k nodes, whose removal gives the highest D_F value, is considered to be the k key nodes for KPP-NEG.

To identify the best set of nodes for each of the above problems, the following procedure (combinatorial optimization) is followed.

Algorithm 1 : Greedy Combinatorial optimization

```
1: Select $k$ nodes at random to populate set $S$
2: Find $F$ = fitness value for the set $S$ using appropriate key node metric
3: for Node $u \in S$ do
4:     for Node $v \notin S$ do
5:         $\Delta$f = improvement in fitness if $u$ and $v$ were swapped
6:     end for
7: end for
8: if $\Delta$f $\leq 0$ then
9:     Terminate
10: else
11:     Swap $u$ and $v$ with greatest $\Delta$f and set $F = F + \Delta$f
12:     Go to step 3
13: end if
```

Information Theory to identify key node sets

Ortiz-Arroyo and Hussain [93] proposed an Information Theory based measure to find *KPP-Pos* and *KPP-Neg* key node sets. This method relies on the structural properties of the network, and uses Shannon's definition of entropy to define the measures.

The connectivity probability distribution of the network is defined as,

$$\chi(v_i) = \frac{deg(v_i)}{2n}$$

Using the above definition, the connectivity entropy H_{co}, is defined as follows,

$$H_{co}(G) = -\sum_{i=1}^{n} \chi(v_i) \times \log_2 \chi(v_i) \tag{2.15}$$

The connectivity entropy measure provides information about the connectivity degree of a node in the graph.

Another probability distribution can be defined in terms of the number of shortest or geodesic paths that have node v_i as source and the rest of nodes in the network as targets. This is called the centrality probability distribution.

$$\gamma(v_i) = \frac{paths(v_i)}{paths(v_1, v_2, .., v_M)}$$

where $paths(v_i)$ is the number of shortest paths from node v_i to all the other nodes in the network and $paths(v_1, v_2, ..., v_M)$ is the total number of shortest paths M that exists across all the nodes in the graph. Using the above definition, the centrality entropy H_{ce}, is defined as follows,

$$H_{ce}(G) = -\sum_{i=1}^{n} \gamma(v_i) \times \log_2 \gamma(v_i) \tag{2.16}$$

Centrality entropy provides information on the degree of reachability for a node in the graph. The algorithm in [93] attempts to solve KPP-POS and KPP-NEG problems using connectivity entropy and centrality entropy. To solve the KPP-POS problem, the set of nodes that produce the largest change in H_{co} is selected. To solve the KPP-NEG problem, the set of nodes that produce the largest change in H_{ce} is selected.

2.2 Key edge identification in networks

Typically, key players in networks refer to important nodes in the network. However in some contexts such as network robustness, edges also play an important role. Although many approaches in the literature have been proposed to identify important nodes, there are very few studies on identifying key edges in networks. Identifying the important edges in a network plays an important role in applications such as the following:

- In information diffusion applications, it is important to identify the edges which play an important role in diffusing information more efficiently among different parts (communities) in the network.

- In determining and strengthening robustness of networked environments, it is important to identify the edges that upon removal would collapse the network, and take necessary measures to protect these edges from attacks and failure.

As mentioned earlier, the number of methods proposed in literature to identify key edges are limited compared to the number of methods proposed in identifying key nodes in networks. In this section, some of the previous work on key edge identification is summarized.

2.2.1 Edge weights

Most of the networks that have been studied are binary in nature; that is, the edges between vertices are either present or not. But some of the networks can also be weighted, meaning their edges can have differing strengths; there may be stronger or weaker social ties between individuals. For example, in a network representing the email exchanges in an office environment, the number of email messages exchanged between two persons can be considered as the edge weight between the corresponding nodes.

As the edges with the highest weights represent the most frequent interactions in the network, edge weights provide a useful means to identify important edges in a weighted network [36].

2.2.2 Edge betweenness

Node betweenness has been studied in the past as a measure of importance of nodes in networks [45, 81]. In order to identify the important edges in a network in terms of appearing in 'between' the shortest paths of pairs of vertices, the node betweenness centrality has been generalized to edges [48]. According to the notation introduced in [48], the edge betweenness centrality for the edge e is defined as,

$$C_{edge\ betweenness}(e) = \sum_{x \neq y \in V} \frac{\sigma_{x,y}(e)}{\sigma_{x,y}} \tag{2.17}$$

where, $\sigma_{x,y}(e)$ is the number of shortest paths between the nodes x and y, that includes edge e, and $\sigma_{x,y}$ is the total number of shortest paths between the nodes x and y. This measure reflects the total number of shortest paths between nodes in the network that rely on a given link. Thus, edges with higher edge betweenness centrality are generally more important for maintaining the connectivity of the network than edges with low centrality.

2.2.3 Edges to improve/reduce robustness

When edges are removed from a social network, the properties of the network change. This amount of change depends on the importance of the set of edges that gets removed. The set of k edges that can reduce the robustness the most upon removal or the set of k edges upon addition that can increase the robustness the most can be considered as 'key' edges in a network. For example, in [21] the set of edges that minimize the natural connectivity of the network [63] upon removal, and the set of edges that maximize natural connectivity upon addition were identified as the important edges in the network. These methods are discussed in detail in Chapter 6, where we focus on improving the network robustness using key edges.

2.3 Concluding Remarks

In this chapter we reviewed some of the previously proposed approaches for key node and key edge identification in network. One underlying property of all the approaches discussed in this chapter is that each of them focuses on one property of interest. But in complicated real life applications, we need a set of key players that can perform well with respect to multiple objectives of interest. To address this problem, in chapter 4, we propose a new algorithm for identifying key players which optimizes multiple objectives of interest.

Chapter 3 Decision making from multi-objective optimization

This chapter introduces evolutionary algorithms and multi-objective optimization, discusses on problems of using multi-objective optimization to real-world decision making. This chapter is organized as follows. Section 3.1 gives an overview of Evolutionary Algorithms (EAs). In Second 3.2, we formally describe the problem of multi-objective optimization. Section 3.3 discusses the problem that multi-objective optimization algorithms produce too many solutions when the number of objectives are high and conflicting, and discusses the approached proposed to solve this problem.

3.1 Evolutionary Algorithms

Various techniques have been proposed to solve optimization problems, and these techniques can be classified into three categories: exhaustive, deterministic and stochastic.

In *exhaustive search*, the entire decision space is searched in order to find the optimal solution. Therefore, exhaustive search techniques are highly computationally expensive and cannot be applied to real world large problems. *Deterministic search* methods incorporate domain knowledge to reduce the size of the search space, which is subsequently probed through tree like or graph like walks. Although the domain knowledge helps to reduce the search space and computational complexity, the domain knowledge is not always available to reduce the search spaces of interest. Evolutionary Algorithms belong to the third category; *stochastic search*, and work by repeatedly sampling the search space, guided by the information collected during the search process. Compared to other methods, Evolutionary Algorithms typically do not attempt to search the entire decision space, and are not guaranteed to find the optimal solution.

Inspired by the natural evolution process [43, 22], Evolutionary Algorithms iteratively modify a population of candidate solutions for the optimization problem. Each solution in the optimization process is referred to as an individual, and through repeated application of randomized processes of recombination, mutation and selection, the individuals are altered until specified termination criteria are met. A typical Evolutionary Algorithm is described by the pseudo-code shown in Algorithm 2 [34].

Algorithm 2 : A typical evolutionary algorithm

```
1:  $i \leftarrow 0$
2:  $P(i) \leftarrow$ Random set of individuals (initial population)
3:  Evaluate the fitness of all individuals of $P(i)$
4:  Choose a maximum number of generations $i_{max}$
5:  while $i < i_{max}$ do
6:      $i = i + 1$
7:      $M(i) = Parent\_selection(P(i-1))$
8:      $C(i) = Offspring\_generation(M(i))$
9:      $P(i) = Select\_for\_survival(P(i-1), C(i))$
10:     Evaluate the fitness of all individuals of $P(i)$
11: end while
12: Return the best individual of $P(i)$
```

Evolutionary Algorithms begin with generating an initial population of individuals drawn at random from the decision space. Then, at each generation i, the mating pool $M(i)$ is generated from the population currently stored as $P(i-1)$.

3.1.1 $Parent\_selection$

During the $Parent\_selection$ stage, an objective function is used to compute the fitness value of each candidate solution, which indicates the quality of the solution. A selection mechanism is then used to select individuals to be used as parents to those of the next generation. Individuals with a high fitness value are given a higher probability to be placed into the mating pool for reproductive purposes. Roulette Wheel Selection[49], Stochastic Universal Sampling[4], Tournament Selection[9] and Truncation Selection[85] are a few of the most common selection techniques used in literature.

3.1.2 $Offspring\_generation$

During the this step, genetic materials between the selected parents are exchanged within the mating pool and it results in the creation of the child population $C(i)$. Offspring generation usually occurs in two forms: crossover and mutation. Every offspring generation operation has an associated probability of occurrence, which is a parameter usually predetermined and kept unchanged throughout the search process.

Crossover or Recombination

The crossover operator is applied to two individuals in the parent population. This creates two new offspring individuals each having different subsets of the alleles of the parents. Figure 3.1 gives an example for one point crossover; a commonly used variant of crossover operator. Here, the point of crossover is determined to be at a particular point, and then the first child inherits alleles (bits) of the first parent upto the crossover point and the alleles of the second parent after the crossover point. The second child inherits alleles of the second parent upto the crossover point and the alleles of the second parent after the crossover point. Other crossover operators have also been proposed in the literature, e.g., two-point crossover and uniform crossover. The new chromosome may be better than both of the parents if it takes the best characteristics from each of the parents.

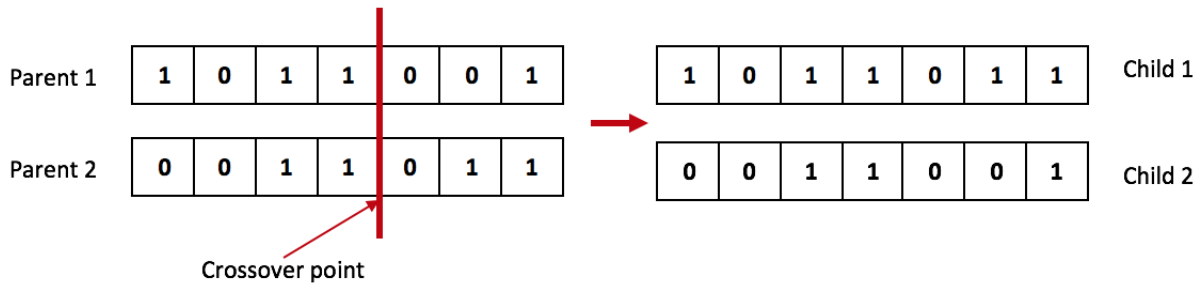


Fig. 3.1: One point crossover operator

Mutation

A mutation operator modifies an individual in the parent population by a slight alteration. Usually this is done by flipping one or more bits of an individual in the parent population to create a new child. Figure 3.2 gives an example of the mutation operator where one bit (5^{th} bit) is flipped to create a new child. Mutation allows the development of un-inherited characteristics in individuals and promotes diversity by allowing an offspring to evolve in ways not solely determined by traits inherited by parents.

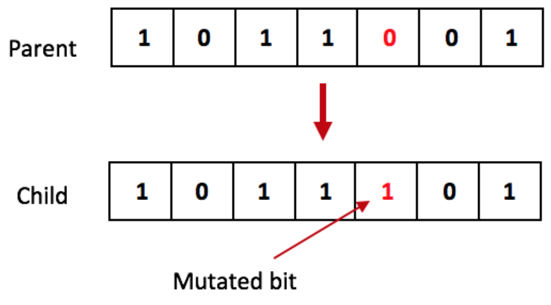


Fig. 3.2: Mutation operator

3.1.3 $Select\_for\_survival$

In this stage, the fitness values of the individuals of the current population $P(i-1)$ and the child population $C(i)$ are compared, and the individuals that form the next generation of the population $P(i)$ are identified. The selection of individuals in this stage does not completely depend on the fitness values. *Elitism* involves using a small fraction of the fittest candidates in the parent population $(P(i-1))$ in the new population $(P(i))$ unaltered, even though fitness values of some of these individuals are less than that of the individuals found after recombination in $(C(i))$ [32]. Elitism avoids the risk of losing highly fit individuals from later generations.

3.1.4 Exploration vs exploitation in evolutionary algorithms

The balance between the *exploration* of unexamined regions of the search space and the *exploitation* of regions already identified as areas containing good solutions plays an important role in evolutionary algorithms [29]. This can be adjusted by modifying the selection pressure implemented using the selection operator and the probability of mutation.

With selection pressure, more emphasis is given to selecting the individuals with high fitness. A strong selection pressure may cause the algorithm to converge rapidly to a local optimum, and a low selection pressure may cause the algorithm to yield random results that differ from one run to another. Crowding[30] is one of the techniques that is widely used to preserve diversity under selection pressure in evolutionary algorithms [79]. There are two main steps involved in crowding. In the pairing step, the offspring individuals are paired with individuals in the parent population according to a similarity metric. In the replacement step, a decision is made for each pair of individuals as to which of them will remain in the population.

Mutation is used to enhance exploration, flipping random bits in an individual. A very high mutation rate increases the probability of searching more areas in search space, preventing the population from converging to optimum solution. On the other hand, a very low mutation rate may result in premature convergence. Hence, the selection of the proper mutation rate for the application at hand is important in determining the performance of the evolutionary algorithm.

3.2 Multi-objective optimization

Multi-objective optimization is the process of optimizing (minimizing or maximizing) a number of objectives simultaneously. A multi-objective optimization problem may also contain a set of constraints which any feasible solution must satisfy. In general, a multi-objective optimization problem can be defined as follows:
Find the vector x^* that optimizes a given set of $\mathcal{O}$ objective functions, i.e.,

$$Maximize/Minimize \;\; F(x^*) = [F_1(x^*), F_2(x^*), ..., F_{\mathcal{O}}(x^*)]^T$$

subjected to the constraints,

$$g_j(x^*) \leq 0 \;\; ; j = 1, 2, ..., k$$

$$h_l(x^*) = 0 \;\; ; l = 1, 2, ..., e$$

Each objective function $F_i(x)$ must be maximized or minimized, $\mathcal{O}$ is the number of objective functions, k is the number of inequality constraints, and e is the number of equality constraints.

In many real-life problems, various objectives conflict with each other. Hence, optimizing with respect to a single objective results in poor solutions with respect to the other objectives. Therefore, in multi-objective optimization we obtain a set of solutions, each of which satisfies the objectives at an acceptable level without being dominated by any other solution.

In multi-objective optimization problems, we expect to find a set of *Pareto-optimal* solutions, each of which is *non-dominated* by any other solution. A solution X is non-dominated, if every solution better than X with respect to one objective function, must be worse than X with respect to another objective function. For example, in a car purchase problem scenario, two objectives of interest to a buyer would be the cost and the engine performance. Alternatives may include a car which has low cost and low engine performance, and another car which comes with high cost and high engine performance. In this scenario, neither is strictly 'better' than the other according to both cost and performance criteria. Such solutions are called Pareto optimal solutions. The set of all possible non-dominated solutions in X is called the *Pareto optimal set*. The corresponding objective function values of the Pareto optimal set in the objective space constitute the *Pareto front*. The goal of a multi-objective optimization algorithm is to identify solutions in the Pareto optimal set.

3.2.1 Evolutionary algorithms for multi-objective optimization

Many different methods exist to solve multi-objective optimization problems. The most common technique is to aggregate the multiple objectives into a single objective by using weighted sum model [86]. Another trivial technique is to optimize the objectives one at the time, with a given order of importance of the objective functions [86]. However, finding the appropriate weight assignment for the objective functions is generally non trivial and problem-dependent. In additions, since these techniques arbitrarily limit the search space some Pareto optimal solutions will not be considered [26].

The application of evolutionary algorithms to solve multi-objective optimization problems is similar to Algorithm 2. However, multi-objective optimization algorithms should also consider how the fitness values should be assigned to individuals to lead the evolution to a Pareto optimal set and how to maintain diversity in the population to avoid premature convergence [34].

Vector Evaluated Genetic Algorithm (VEGA) [100], was the first evolutionary algorithm proposed to solve multi-objective optimization. During each generation of VEGA, a number of sub-populations are generated by performing proportional selection according to each objective function. Then these sub-populations are shuffled, and regular GA operations are carried out on the shuffled populations. The concept of Pareto optimality was introduced by David E. Goldberg in [50], and has been used by almost all the evolutionary algorithms proposed to solve multi-objective optimization afterwords.

Non-dominated Sorting Genetic Algorithm (NSGA) was proposed by Srinivas and Deb [106], and is based on several layers of classification of the individuals. The key steps of the NSGA are as follows:

i. Before selection, the population is ranked based on non-domination, where all non-dominated individuals are given the same rank.

ii. All these individuals share the same fitness value.

iii. Then, this group of individuals are ignored and the next set of non-dominated individuals are obtained from the remaining layers.

iv. These individuals are given a fitness value less than that of the previous set.

v. The process continues until all individuals in the population are assigned a rank.

Since the individuals in the first non-dominating set have the highest fitness value, more individuals of that set get selected to the mating pool. NSGA was shown to be a computationally expensive algorithm for multi-objective optimization because of repeated calculation of non-dominating sets [27]. Niched-Pareto Genetic Algorithm (NPGA) [56], which uses a tournament selection scheme based on Pareto dominance, and Multi-Objective Genetic Algorithm (MOGA)[44], which ranks individuals based on the number of other individuals which are dominated by it, were also proposed during the same period. Strength Pareto Evolutionary Algorithm (SPEA) [127] uses a generational gap elitist approach,

Algorithm 3 : The NSGA-II algorithm

```
1: $i \leftarrow 0$
2: $P_i \leftarrow$ Random set of individuals (initial population) of size $N$
3: Evaluate the fitness of all individuals of $P_i$
4: Apply crossover and mutation to $P_i$ to create offspring population $C_i$ of size $N$
5: Choose a maximum number of generations $i_{max}$
6: while $i < i_{max}$ do
7:     Set $R_i = P_i \cup C_i$
8:     Identify the non-dominated fronts $F_1, F_2, ..., F_k$ in $R_i$.
9:     for $j = 1, ...k$ do
10:        Calculate $crowding\ distance$ of the solutions in $F_i$
11:        $P_{i+1} = \emptyset$
12:        if $(|P_{i+1}| + |F_j| \leq N)$ then
13:            $P_{i+1} = P_{i+1} \cup F_j$
14:        else
15:            Add the least crowded $N - |P_{i+1}|$ individuals from $F_j$ to $P_{i+1}$
16:        end if
17:    end for
18:    Use binary tournament selection based on the crowding distance to select parents
       from $P_{i+1}$
19:    Apply crossover and mutation to $P_{i+1}$ to create offspring population $C_{i+1}$ of size N
20:    $i \leftarrow i + 1$
21: end while
```

where a proportion of the current population is preserved and carried to the next generation. Strength Pareto Evolutionary Algorithm 2 (SPEA2) [126] is an improved version of the SPEA.

The most widely used evolutionary algorithm for multi-objective optimization is the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [31], which is an improved version of NSGA [106], and is shown in Algorithm 3.

NSGA-II estimates the density of solutions surrounding a particular solution in the population by computing the average distance of two points on either side of this point along each of the objectives of the problem. This value is called the $crowding\ distance$. During selection, the NSGA-II considers both the non-domination rank of an individual and its crowding distance. The elitist mechanism used in NSGA-II consists of combining the best parents with the best offspring obtained. NSGA-II is much more efficient than its predecessor, and the superior performance is evident from the wide usage of the algorithm in wide range of applications [70].

Some of the more recent evolutionary algorithms proposed to solve multi-objective optimization problems includes MOEA/D [125], BORG[53] and NSGA-III [33, 61].

As NSGA-II has been widely used and shown superior performance in multiple applications, we use NSGA-II as the multi-objective optimization algorithm in this study.

3.3 Large number of solutions in $\mathcal{O}$-objective optimization

One issue with regard to $\mathcal{O}$-objective optimization is that we obtain a large number of solutions when the value of $\mathcal{O}$ is high and the objectives are uncorrelated [46, 41, 7]. But, the decision makers who use multi-objective optimization in their applications usually require one or two solutions to be used in their applications. Multiple methods have been proposed in literature to prune the Pareto optimal set of solutions. This section discusses some of the methods proposed in literature.

3.3.1 Selecting solutions from the Pareto optimal set

The methods proposed to select solutions from the Pareto optimal set can be divided into three categories.

Ranking methods

In ranking methods, after executing the multi-objective optimization algorithm, the set of Pareto optimal solutions obtained are ranked according to a user-specified certain criteria. Once the ranking is done, the decision maker can pick the solutions that are best ranked for the desired applications. Some of the proposed ranking methods include the following:

1. *Weighted sum approach (WS)* :

   This is the most widely used approach for pruning solutions from the Pareto optimal set. For an $\mathcal{O}$-objective optimization problem, the weighted sum rank of the Pareto optimal solution X_i is given by,

$$WS(X_i) = \sum_{j=1}^{\mathcal{O}} w_j O_j(X_i), \tag{3.1}$$

   where w_j is the weight assigned to the objective O_j. The weight assignment to the objectives is domain dependent and the decision maker should determine the appropriate weight assignment to the objectives. The result of the ranking depends on the weight assignment. Hence, in applications where the proper weight assignment is unknown, the results of the weighted sum approach are questionable [46].

2. *Average Ranking (AR)*:

   This method uses the average of the ranking positions of a solution X_i given by all the objective functions, and is calculated as follows:

$$AR(X_i) = \frac{\sum_{j=1}^{\mathcal{O}} R_j(X_i)}{\mathcal{O}}, \tag{3.2}$$

   where $R_j(X_i)$ is the rank given to the solution X_i by the objective O_j.

3. *Maximum Rank (MR)*:

   This approach does not assign a rank to each of the solutions in Pareto set. The main steps of the MR are as follows,

   i. Solutions in the Pareto set are ranked separately for each objective.

   ii. The best ranked k points from each objective are extracted.

   As this approach selects the best solutions for each objective independently, this tends to extract solutions from extreme points in the Pareto surface [120].

Pruning methods

The solution pruning methods proposed in the literature can be divided into two categories.

1. Clustering:

   The clustering method assumes that the output of the pruning process should be the distinct solutions in the objective space. The number of clusters can either be determined by the decision maker or can be optimized according to the Pareto set solutions. For each cluster, one representative solution is chosen in which often the solution nearest to the center of the cluster is used. The number of clusters is optimized using the average silhouette width [98]. For a solution X_i, this approach calculates the average distance $a(X_i)$ to all other points in its cluster and the average distance $b(X_i)$ to all other points in the nearest neighbor cluster.

   $$Silhouette(X_i) = \frac{b(X_i) - a(X_i)}{max(a_i, b_i)} \tag{3.3}$$

   A silhouette value close to 1 indicates that the solution was assigned to an appropriate cluster. If the silhouette value is close to 0, it means that the solution could be assigned to another cluster; and if it is close to -1, the solution is considered to be misclassified. The overall silhouette width is the average of the silhouette values of all solutions. The largest silhouette width indicates the best clustering and therefore the number of clusters associated with this best clustering is taken as the optimal number of clusters. The following are two approaches used to select representative points from the clusters.

   (a) *Cluster centers (CC)*:

   In this method, after the clustering algorithm is executed, the centroids of the clusters are chosen as the representative points from each cluster. In [23], k-means [54] is used as the clustering algorithm, and the cluster centroids are picked as the representative points.

(b) *Points closest to the Ideal point (IP)*:

The main steps of this approach are as follows [24].

i. For each cluster, the ideal point is identified. The ideal point of a subset of points is the virtual point that has a minimal evaluation for each objective.

ii. Then, for each point in each cluster, the distance from to the ideal point of the cluster is calculated.

iii. From each cluster, the point with the smallest distance to the ideal point is selected.

However, clustering methods do not necessarily guarantee an even spread of solutions, as they are sensitive to the presence of outliers. Also, in cases where the Pareto optimal set does not form any clusters, identifying solutions based on clustering is not ideal.

2. Angle based pruning:
   In this method, the geometric angle between each pair of solutions is calculated, for each objective. A threshold angle is defined for each objective, in order to identify the subset of desirable solutions. The idea is to remove the solutions that only improve some objectives marginally while significantly worsening other objectives [108]. This method may identify the knee points [6] in the Pareto set.

Subset optimality

Each point in the Pareto optimal set is non-dominated by any other point in the same Pareto optimal set with regard to the $\mathcal{O}$ objectives on which the multi-objective optimization algorithm is run. But, when a subset of the $\mathcal{O}$ objectives is considered, some of the points in the Pareto optimal set may dominate other points. Some methods have been proposed to use the concept of subset optimality to reduce the number of solutions in the Pareto optimal set. Some such methods include:

1. *Favor Relation (FR)*:

   The favor relation [35, 28] is defined as, 'a solution X_i is favored over the solution X_j if and only if X_i is better than X_j on more objectives than X_j on X_i'. Depending on the favor relation between the solutions of the Pareto set, the following steps are followed to create a directed network and prune the Pareto set.

   i. If X_i favored over X_j, an edge from the node X_i to X_j is created.

   ii. Collapse all the nodes in each cycle to a single pseudo node (The favor relation may not be transitive, thus the network may have cycles). Each node inside a pseudo-node is not better than another in the same cycle.

   iii. The nodes with $in\_degree = 0$ are selected.

   As cycle identification is computationally expensive, there are computational limitations in applying this algorithm to Pareto sets that create large directed networks.

2. *K-optimality (KO)*:

   The concept of k-optimality was introduced in [34], and was used to prune solutions from the Pareto optimal solutions. A point X_i in a set of non-dominated $\mathcal{O}$ objective points is efficient with order k, where $1 < k < \mathcal{O}$, if and only if X_i is non-dominated in every k objective subset of the $\mathcal{O}$ objectives. The points that show the highest order k optimality are selected from the Pareto optimal set.

One issue with the all aforementioned methods for pruning Pareto optimal solutions is that these algorithms need to be run after $\mathcal{O}$ objective optimization is completed. Hence the decision makers have to incur more computational cost in addition to the computational cost of $\mathcal{O}$ objective optimization algorithm. We propose an algorithm in Chapter 4 which not only reduces the number of solutions in the Pareto set but also reduces the computational cost compared to previously proposed algorithms.

3.4 Concluding Remarks

Evolutionary algorithms have been widely used to solve multi-objective optimization problems. One prevailing problem with multi-objective optimization algorithms is that they produce too many solutions when the number of objectives are high and conflicting. But the decision-makers who use multi-objective optimization algorithms in their applications require one or two solutions to be used in their applications. Multiple methods have been proposed to select solutions from the Pareto optimal set, but these need to be invoked after the multi-objective optimization algorithm is executed. Hence the decision-makers have to incur more computational cost in addition to the computational effort required by the multi-objective optimization algorithm.

CHAPTER 4
IDENTIFYING MULTI-OBJECTIVE KEY NODES

In Section 2.1 we presented the existing methods to identify the key nodes in a network. One underlying property of all the measures presented in Section 2.1 is that each of them focus on one property of interest. For example, the key players identified by eigenvector centrality are the nodes well connected to important nodes in the network, and the key players identified by closeness centrality are the nodes that are in the center of the network. But, for most of the real world applications, we need a set of key players who can perform well on multiple objectives of interest. For example, in selecting a set of seeds for an application of information propagation, we would ideally need a set of nodes which can reach all the nodes in the network quickly (high closeness centrality), and are also connected to the more important nodes (high eigenvector centrality). Since, each existing algorithm for key node identification only focuses on one objective of interest, these algorithms cannot find key players who can perform well on multiple objectives of interest in many applications. Also, the 'collective' behavior of key nodes is ignored in existing key node identification methods.

This chapter is organized as follows. In Section 4.1 we introduce some of the defi-

Table 4.1: Statistics of the networks used

Network	Nodes	Edges	Description
Dolphin[1]	62	159	Frequent associations between dolphins that lived off Doubtful Sound, New Zealand
Prisoners[2]	67	142	Sociometric choice data collected from 67 prison inmates

ciencies in current key node identification methods. Then, to address these deficiencies, we introduce the idea of 'identifying multi-objective key nodes' in Section 4.2. Then in Section 4.3, we show how the multi-objective key node identification method solves one of the deficiencies identified in the current approaches. Finally Section 4.5 compares the different key node identification methods in two well known applications and show that the multi-objective approach outperforms the existing key node identification methods.

4.1 Deficiencies of current approaches for key node identification

In this section we discuss a few deficiencies found in single objective key node identification algorithms. For illustrative purposes we use Dolphin and Prisoners datasets which are publicly available in UCI Network Data Repository and Table 6.1 shows the statistics and descriptions of the two networks.

For example, the Dolphin social network [77] represents the frequent associations between dolphins in a community living off Doubtful Sound, New Zealand. Figure 4.1 shows the network structure and the communities detected using the modularity optimization algorithm proposed by Blondel, et al.[10]. Sizes of the nodes are proportional to Eigenvector centrality, and different communities are denoted by different colors.

[1]Source: http://networkdata.ics.uci.edu/data.php?id=6

[2]Source: http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/ucidata.htm#prison

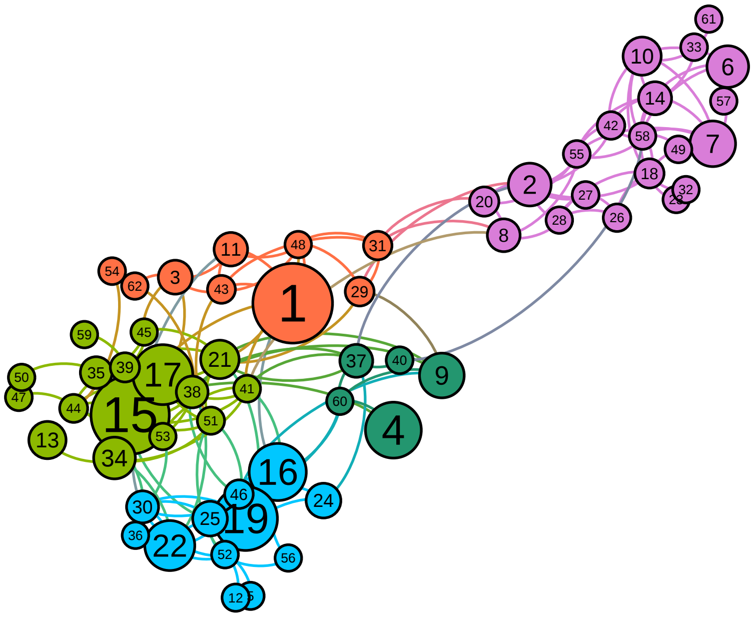


Fig. 4.1: Dolphin Network

4.1.1 Collective behavior of a set of key nodes

One problem with previous approaches for key player identification algorithms is the influence-overlapping of the key players that these algorithms identify. In other words, one key player's contribution may overlap with the contribution of another key player. A 'good' key player identification algorithm should identify k key players who can 'collectively' perform well.

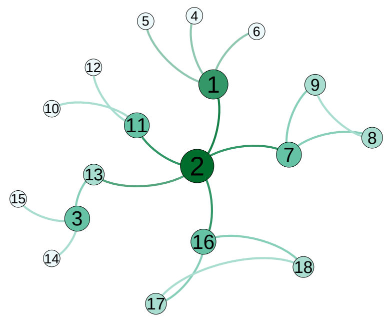
To measure the collective influence of a set S of k key players, we follow the *minimum method* introduced in [40]. The minimum method was introduced to capture the behavior of a group of nodes, once formed, need to act as a single unit [40]. To model the collective behavior of a set of nodes S, the concept of 'super' node S_{super} is used.

Given the network $G = (V, E)$ and a set of nodes S, Algorithm 4 generates a new network $G' = (V', E')$, which consists of S_{super}. In G', the set of nodes S, is replaced by a single node S_{super} and all the neighbors of the set of nodes S are connected to S_{super}.

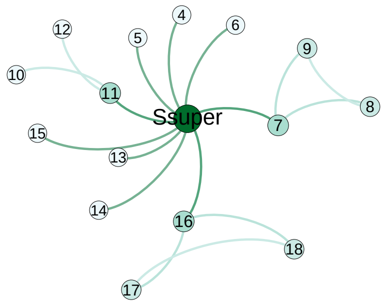
Algorithm 4 : *GetSuperNode*: The algorithm to create S_{super}

```
Input: Set of nodes $S$, Network $G = (V, E)$
Output: Network $G' = (V', E')$, Super Node $S_{super}$
 1: $N_s = \{\}, E' = E, V' = V \cup \{S_{super}\}$
 2: for all Node $j \in S$ do
 3:     $N_j = \{u \mid u \in V$ and $(j, u) \in E\}$
 4:     $N_s = N_s \cup N_j$
 5:     $E' = E' \setminus \{(j, v') \in E' \mid v' \in N_j\}$
 6:     $V' = V' \setminus \{j\}$
 7: end for
 8: for all Node $i \in N_s$ do
 9:     $E' = E' \cup (S_{super}, i)$
10: end for
11: return $G' = (V', E')$, $S_{super}$
```

Figure 4.2 shows an example for the S_{super} node creation. Initial $G = (V, E)$ is shown in Figure 4.2(a) and $S = \{1, 2, 3\}$. Figure 4.2(b), shows network $G' = (V', E')$ after the creation of the node S_{super}. Finally, the collective centrality measure of S is the measure associated with S_{super} in the new network $G' = (V', E')$.



(a) Original Network $G = (V, E)$ and S=(1,2,3)



(b) Network $G' = (V', E')$, with created node S_{super}

Fig. 4.2: Creation of the S_{super} as a node for a set of nodes S

To capture the collective behavior of the sets of 5 key players identified by Eigenvector, Betweenness and Closeness centrality measures, we follow these steps:

i. Identify the set of 5 top key players using each centrality measure.

ii. Use Algorithm 4 to generate the 'super' node S_{super} for each top set of 5 top key players.

iii. Calculate the appropriate centrality measure for the generated 'super' node S_{super}.

In order to identify the set of 5 key nodes which gives the best collective behavior for each centrality measure, we use a Genetic Algorithm based optimization algorithm.

The set of 5 top key players identified for Dolphin and Prisoners networks are shown in the 3^{rd} column of Table 4.2 and the appropriate centrality values of the 'super' node S_{super} created by each of these set of 5 top key players are shown in the 4^{th} column of Table 4.2. The sets of 5 nodes (which optimizes each centrality value for Dolphin and Prisoners networks) are shown in the 5^{th} column of Table 4.2 and the appropriate centrality values of the 'super' node S_{super} created by each of these set of 5 nodes are shown in the 6^{th} column of Table 4.2.

Table 4.2: Collective behavior of the key players

Network	Centrality Measure	Top 5 key players	Collective centrality value	Best top 5 nodes for collective behavior	Collective centrality value
Dolphin	Eigenvector	1, 15, 19, 17, 16	0.513	34, 15, 58, 46, 21	**0.552**
	Betweenness	38, 44, 21, 15, 30	0.579	8, 15, 40, 38, 16	**0.697**
	Closeness	15, 11, 9, 16, 14	0.528	15, 2, 37, 19, 17	**0.740**
Prisoners'	Eigenvector	51, 36, 40, 29, 54	0.548	40, 7, 28, 54, 15	**0.596**
	Betweenness	15, 7, 29, 54, 51	0.744	7, 29, 40, 46, 51	**0.759**
	Closeness	51, 29, 15, 36, 7	0.585	7, 13, 36, 46, 55	**0.667**

It can be seen from the results in Table 4.2 that none of the 5 top key nodes identified by any of the centrality values could achieve the best collective behavior. This behavior is further evident from the poor performance of single objective key player algorithms in the results shown in Chapter 4.

4.1.2 Optimization of a single property

Although many algorithms have been proposed to identify a key node or a set of key nodes, they all share a common characteristic. All the aforementioned approaches define ‘key nodes’ with an appropriate objective of interest, and find sets of key nodes which optimize the identified objective. We call such algorithms as ‘single objective’ key node identification algorithms in this study. Consequently, the prevailing deficiency of each of these methods is that, they perform well only when we consider their objective of interest as the only characteristic that the set of key players should have. But in complicated real life applications, we need a set of key players which can perform well with respect to multiple objectives of interest.

Let us consider Eigenvector centrality as an example. Eigenvector centrality gives priority to nodes that are connected to other important nodes. One known deficiency of the Eigenvector centrality approach is that it tends to find key nodes that are all within the same region of a network [60]. When key nodes are identified in a massive social network with multiple communities, ideally the key nodes should represent all the communities in the network. A strong argument in favor of identifying key nodes from different parts of a given network was given by Granovetter [52]. In this paper, Granovetter argues that members of one community have much to gain from acquaintances (nodes belonging to other communities) for fresh ideas. So, for a set of key nodes to have diverse ideas and to represent the ideas of the whole population, they should represent all parts of the population. Another example for the need of identifying key nodes from different parts of a network is target marketing. In this case it is important to target the ‘right’ set of key people in a population to spread information efficiently and effectively. If the set of initial seeds (key nodes) for the information were identified from the same community in a network, the information spread will be limited to that particular community.

In an attempt to cure this weakness Ilays and Radha introduced Principal Component centrality [60], but this method was also unable to capture the key concerns raised above when applied to some social networks, and results in finding key players from a small number of communities.

We applied Eigenvector centrality and Principal Component centrality to identify key nodes in the Dolphin network. According to Eigenvector centrality, the set of 5 key nodes was found to be {1, 15, 19, 17, 16}, and all 5 key nodes come from only three communities in the network. Similarly, the set of 5 key nodes {1, 15, 4, 9, 21} identified by Principal Component centrality also belong to only three communities. Similarly, the set of 5 key nodes identified on the Prisoners network ({54, 55, 48, 63, 51}) represent only two communities.

Eigenvector centrality manages to identify a set of key nodes connected to important nodes in the network, but ignores importance of the distribution of the key nodes as we just observed. Such a deficiency is not unique to Eigenvector centrality approach; other key node identification algorithms that focus on single objective optimization can suffer also from this problem. In Section 4.3 we show how this problem can be addressed using the multi-objective approach that we propose in this thesis.

4.2 Multi-objective optimization for identification of k key nodes in social networks

The set of key players that we identify must possess multiple important properties of interest. To simultaneously optimize all the relevant objectives for a certain application, we model the key player identification problem as a multi-objective optimization problem. In this case, the objective functions describe the set of properties the key players should possess.

A binary representation of the network is used in a Genetic Algorithm (GA). In the binary representation, each node in the network is assigned an index. The number of bits in the bit string is equal to the number of nodes (n) in the network. Initially, before selecting any key players, the bit string consists of all 0s. When a certain node is selected to be a key player, the bit value corresponding to the index of the selected node will be changed to 1. For example, if nodes 4, 6, 10, 14 and 19 were selected as key nodes from a social network of 20 nodes, this will be represented as a bit string of length 20 with indices 4, 6, 10, 14 and 19 selected as 1s and rest as 0s. The key steps of the Genetic Algorithm are:

i. Initial population - In each individual (bit string) in the initial population, k random bits are assigned the value of 1 to indicate that they are selected as key players, and the remaining $(n - k)$ bits are assigned 0.

ii. Fitness values - The fitness value of each selected set of k key nodes is calculated by constructing the node S_{super} (using Algorithm 4 in Section 4.1.1) and evaluating the multiple desired properties of interest for S_{super}.

iii. Crossover - Crossover is applied to a fraction P_c of selected individuals to generate offspring.

iv. Mutation - Mutation is performed by flipping each bit value with a probability P_m.

$P_c = 0.9$ and $P_m = 0.1$ are used in the experiments in this study. After crossover and mutation, the resulting bit strings are readjusted to contain k 1s and $(n - k)$ 0s by randomly adding or deleting some 1s as the case requires.

As mentioned earlier, each individual is evaluated on multiple properties of interest and the individuals that optimize all the properties considered simultaneously are identified by the multi-objective optimization. Once the optimization completes, we obtain a Pareto surface of non-dominated solutions. The number of solutions obtained depends on the network on which the optimization is performed and the objectives selected.

4.3 Addressing the deficiency of Eigenvector Centrality using Multi-Objective Optimization

We now consider the problem of Eigenvector centrality that we discussed in Section 4.1.2. Suppose we need to find 5 key nodes in this case. The issue was that the identified key players were too close to each other.

4.3.1 Using community information as an objective

One possible way to solve this issue of Eigenvector centrality is by using the community information in the multi-objective optimization. In this case, in addition to maximizing the Eigenvector centrality of the super node constructed by the key nodes, a second objective of maximizing the number of communities represented by the set of key players was introduced into the problem. The idea here is that, when the number of communities represented by the set of key players increases, the set of key players would spread out in the network.

When the above problem formulation is compared with the description of multi-objective optimization provided in Section 3.2, the Eigenvector centrality and the number of communities represented by the set of key players are the two objective functions ($F_i(x)$) considered. The equality constraint ($h(x)$) is that the number of key players selected is equal to 5.

Tables 4.3 and 4.3 show the results obtained from using multi-objective optimization algorithm for key node identification. As expected, there are multiple non-dominating solutions for each network. For example, from the results obtained for the Dolphin network, the solution which represents least number of communities has the highest collective Eigenvector centrality value and the solution which represents highest number of communities has the lowest collective Eigenvector centrality value. The solution {48, 58, 37, 46, 15}, represents all the communities in the network.

One could argue that identifying nodes with the highest Eigenvector centrality from

Table 4.3: Set of multi-objective key players found for Dolphin Network. Objectives : Eigenvector centrality (EC) of the super node, Number of communities represented by the key nodes

The set of key players	EC of the super node	Number of communities represented
34, 15, 58, 46, 21	0.552	3
48, 58, 38, 46, 15	0.549	4
48, 58, 37, 46, 15	0.541	5

each community separately also can solve this problem without using multi-objective optimization. But the solutions obtained from this approach have low collective Eigenvector centrality value compared to the results obtained in multi-objective approach. For example, in the Dolphin network, the super node constructed by the solution obtained from identifying nodes with the highest Eigenvector centrality from each community separately ({4, 7, 15, 19, 1}) has an Eigenvector centrality of 0.49, which is lower than the Eigenvector centrality of the super nodes constructed from the solutions obtained from the multi-objective approach.

4.3.2 Using distance as an objective

If the communities in the network are not known, the second objective of maximizing the distance between the key players can be introduced into the problem. The reasoning behind distance maximization is to spread out the set of identified key players. The intention is to find solutions (sets of key players) that maximize both objectives.

Figure 4.3(a) shows the Pareto optimal front identified for the Dolphin network and Figure 4.3(b) shows the same for the Prisoners network [78], where

Table 4.4: Set of multi-objective key players found for Prisoners Network. Objectives : Eigenvector centrality (EC) of the super node, Number of communities represented by the key nodes

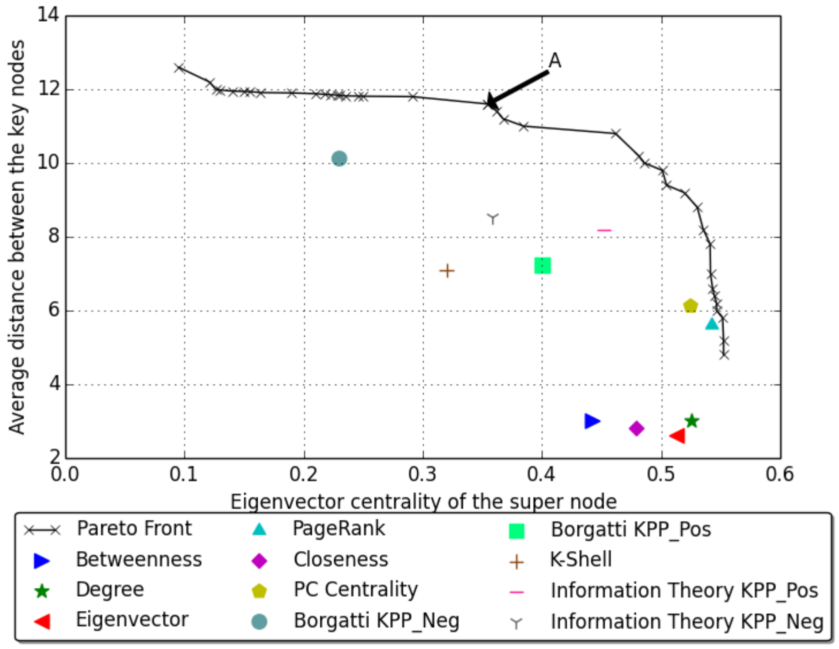
The set of key players	EC of the super node	Number of communities represented
40, 7, 28, 54, 15	0.596	3
40, 7, 36, 54, 15	0.592	4

i. the x-axis represents the Eigenvector centrality of the S_{super} node created from the selected set of five key players, and

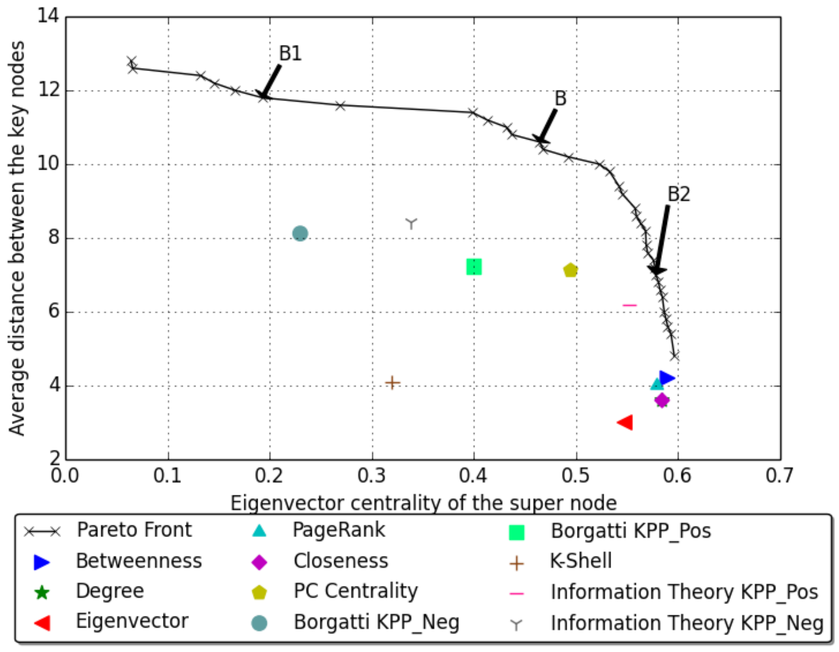
ii. the y-axis represents the average distance between the selected set of key players.

As mentioned earlier, all the points in the Pareto front are non-dominated thus, depending on the importance given to each objective function, all points in the Pareto front provide a set of 5 key players. For example, point A in the Pareto front shown in Figure 4.3(a) refers to the set of key players $\{1, 17, 22, 9, 7\}$ and they belong to 5 different communities in the network. More concretely, consider Figure 4.3(b), the Pareto optimal front for Prisoners Network. In this figure point B refers to the set of five key players {4, 14, 29, 54, 63} that belong to 4 different communities in the network. We now consider two additional points in the Pareto front: $B1$ and $B2$. Although these 3 solutions are non-dominated and intended to optimize both objectives, they assign different weights to the two objectives considered here. The point $B1$ which refers to the key player set {0, 10, 18, 37, 51}, has a high distance between the selected key players, but its average Eigenvector centrality is low compared to B and $B2$. This indicates that the solution $B1$ is appropriate if a higher weight should be given to the distance between the key players, as opposed to the average Eigenvector centrality. On the other hand, the point $B2$ which refers to the key player set {7, 36, 40, 51, 54}, has a high average Eigenvector centrality but low distance between the key players compared to $B1$ and B. If one intends to find key players which give high significance to the average Eigenvector centrality as opposed to the distribution of the key players, $B2$ is a better choice than B and $B1$. Compared to point $B1$ and $B2$, the key player set identified by point B, gives equal weight to average Eigenvector centrality and distribution of the key players. The selection of ideal key player set from the suggested points in the Pareto front is application oriented.

The same principle can be used to optimize other objectives as well, such as Borgatti's positive and negative KPP. The idea is to identify key players who are optimally connected to the rest of the network and will maximally disconnect the network upon deletion. Figure



(a) Dolphin Network, Objectives - Eigenvector centrality of the super node and average distance between key players



(b) Prisoners Network, Objectives - Eigenvector centrality of the super node and average distance between key players

Fig. 4.3: Pareto Fronts : Objectives - Eigenvector centrality of the super node and average distance between key players

4.4(a) shows the Pareto front generated for the Dolphin Network for this case. For example point C in the Pareto front refers to the set of key players {2, 31, 16, 21, 1}. Figure 4.4(b) shows the Pareto front generated for Prisoners Network considering the same two objectives where point D in the Pareto front refers to the set of key players {7, 15, 46, 51, 60}.

4.4 Selection of key players sets

Multi-objective optimization identifies multiple sets of solutions which fall on the Pareto front. For example, as shown in Figure 4.3(a) and 4.3(b), when we use the objectives 'Eigenvector centrality of the super node' and 'distance between the key players', Dolphin Network and Prisoners network have 56 and 44 points in the Pareto front, respectively.

In Chapter 4, we discussed a few approaches that have been proposed to select solutions from the Pareto set. But all those approaches require computational cost in addition to the computational cost of $\mathcal{O}$ objective optimization algorithm. We propose an algorithm, viz., *Leave-k-out* approach, which not only reduces the number of solutions in the Pareto set but also reduces the computational cost compared to previously proposed algorithms.

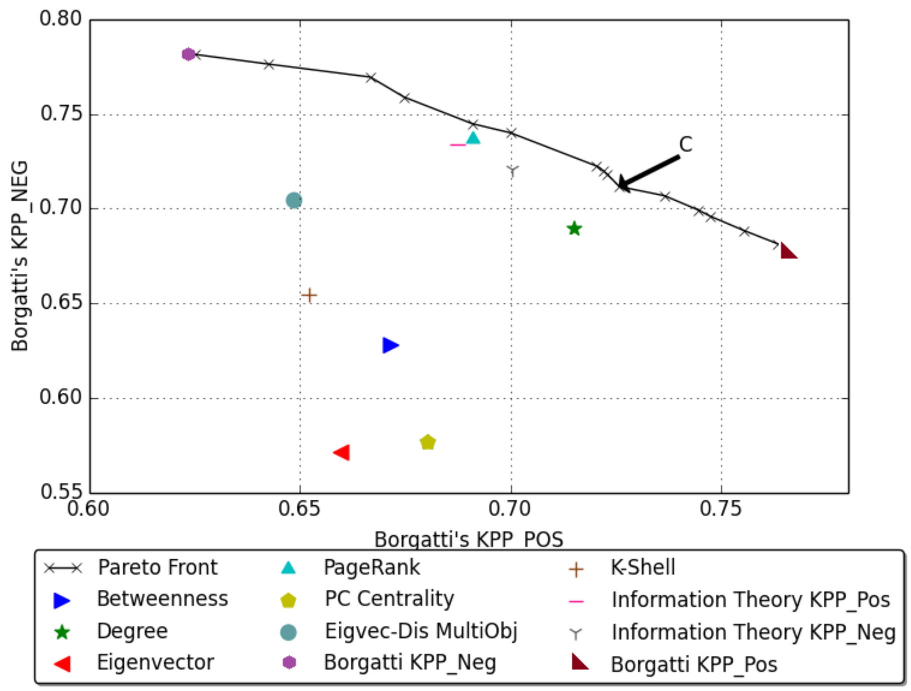
4.4.1 *Leave-k-out* approach for multi-objective optimization

The *Leave-k-out* approach for an $\mathcal{O}$ objective optimization problem is described below:

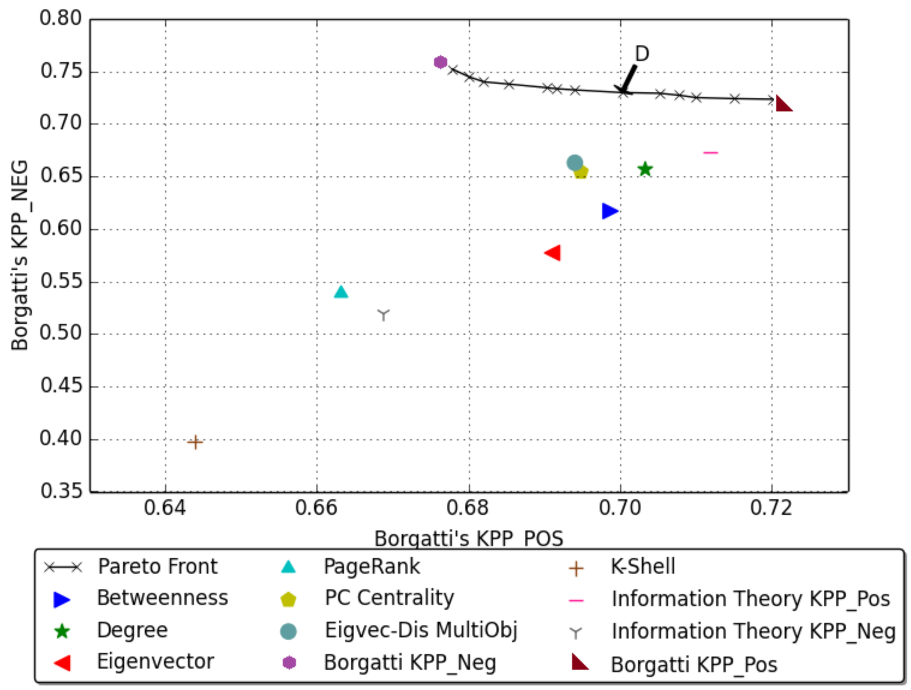
i. Select $(\mathcal{O} - k)$ objectives from the set of objectives and run the multi-objective optimization algorithm.

ii. Obtain the Pareto set, and evaluate each solution in the Pareto set on the objectives that were left out.

iii. Select the solutions in the Pareto set which are non-dominated on the evaluation of the objectives which were left out.



(a) Dolphin Network, Objectives - Borgatti's KPP POS and Borgatti's KPP NEG



(b) Prisoners Network, Objectives - Borgatti's KPP POS and Borgatti's KPP NEG

Fig. 4.4: Pareto Fronts : Objectives - Borgatti's KPP positive and negative

Compared to the other approaches proposed, the *Leave-k-out* approach has the following advantages.

1. A high percentage of solutions obtained are a subset of the Pareto surface obtained by $\mathcal{O}$ objective optimization.

2. The running time of the optimization reduces, as $(\mathcal{O} - k)$ objective optimization requires less computational effort than the original $\mathcal{O}$ objective optimization problem.

3. This method does not require any additional processing such as ranking, clustering etc. after the Pareto set identification, unlike the other approaches.

In using the *Leave-k-out* to select the best k key player set, we assume that the selected set of key players are needed to perform well “collectively" (as a single unit). For example, if a set of k key players were picked to initiate a marketing campaign, these k key players should perform well collectively to spread information effectively.

Let $\mathcal{A} = \{a_1, a_2, ..., a_N\}$ be the complete set of qualities (objectives) that can be used to identify a set of k-key players. The items in the set $\mathcal{A}$ are the measures, such as average Degree centrality of the key players, average Eigenvector centrality of the key players, Borgatti’s KPP Positive measure, Borgatti’s KPP Negative measure, etc., as discussed in Section 2.1. Let $\mathcal{M} = \{m_1, m_2, ..., m_n\} \subseteq \mathcal{A}$ be the set of properties measuring the qualities desirable for the key nodes, as determined by the application of interest, such as a target marketing campaign. The subset of measures used to identify key players in a political campaign may be different from the subset of measures picked to identify key players for a marketing campaign.

In the previous example (addressing the deficiency of Eigenvector centrality), two qualities are chosen from $\mathcal{M}$ to identify sets of non-dominated k key players. When 2 qualities were chosen in the initial step to identify the Pareto front, it helps us better visualize and compare other key node identification methods with the multi-objective approach of key node identification. Now we use the remaining qualities in $\mathcal{M}$ to select a smaller subset from the Pareto front, as described below. Since we assume that the set of k key nodes should “perform" well as a single unit, we represent the set of key nodes as a single super-node as discussed in Algorithm 4 in Chapter 2. Then, we evaluate the “performance" of this super-node, in terms of other measures in $\mathcal{M}$ which are not used to draw the Pareto front. Finally, we restrict attention to non-dominated vectors to find the desired set of key players. Algorithm 5 describes this approach.

Algorithm 5 : Reducing the number of Key Player sets

```
Input: Sets of key players found in the Pareto Front ($S$), Network $G = (V, E)$, $\mathcal{M}' = \{m_{l+1}, m_{l+2}, ..., m_n\}$
Output: Sets of key players ($T$, where $|T| \leq |S|$)
 1: for all set of key players $s \in S$ do
 2:     $m_{l+1}(s) \leftarrow 0, m_{l+2}(s) \leftarrow 0, m_{l+3}(s) \leftarrow 0, ..., m_n(s) \leftarrow 0$
 3: end for
 4: for all set of key players $s \in S$ do
 5:     $(G' = (V', E')$, $S_{super}$ ) = $GetSuperNode(G = (V, E), s)$
 6:     $i \leftarrow l + 1$
 7:     for $i \leq n$ do
 8:         $m_i(S_{super}) \leftarrow Evaluate(m_i, S_{super})$
 9:         $m_i(s) \leftarrow m_i(S_{super})$
10:     end for
11: end for
12: $T \leftarrow Find\_non\_dominated\_sets(m_{l+1}, m_{l+2}, m_{l+3}, ..., m_n)$
13: return $T$
```

Assume we initially use l objectives first (of the set $\mathcal{M}$) to construct the Pareto front. Inputs to Algorithm 5 are, (i) the set $\mathcal{M}' = \{m_{l+1}, m_{l+2}, m_{l+3}, ..., m_n\}$ (the k objectives that were left out) (ii) the set S of k-key players found by the Pareto front using two selected objectives from the set $\mathcal{M}$, and (iii) the network $G(V, E)$. A super node is constructed to represent each set of key players as a single node using the function ‘$GetSuperNode$’ introduced in Algorithm 4. The objectives in $\mathcal{M}'$ are used to evaluate the super node S_{super} using the function $Evaluate(m_i, S_{super})$, and the set(s) of non-dominated key players is selected.

Now we consider how this algorithm can be applied to the Pareto fronts obtained by the Dolphin and Prisoners networks. To obtain the Pareto front we used two properties from

Table 4.5: Performance Criteria and Measures for sets of Key Players

Performance Criteria	**Measured By**
Directly connected to as many nodes as possible	Degree centrality
Should be able to mediate communication between communities	Betweenness centrality
Should be able to communicate quickly with all the nodes	Closeness centrality
Should be connected to important nodes	PageRank

the set $\mathcal{M}$, namely, eigenvector centrality and the distance between the key nodes. For this application, suppose set $\mathcal{M}'$ consists of measures mentioned in Table 4.5 and each set of five selected key players is required to do well with respect to all four capabilities. A fraction of the sets of key players, suggested by the Pareto front, is presented in Table 4.6 for the Dolphin Network and in Table 4.7 for the Prisoners Network. DC, BC, CC and PR stand for Degree centrality, Betweenness centrality, Closeness centrality and PageRank respectively. The set of players identified by Algorithm 5 is depicted (in bold) in both tables. The sets selected by Algorithm 5 are non-dominated and the users can select any set depending on the requirements. Both examples illustrate that the algorithm significantly reduces the desired set of key players (from 56 to 3 in the Dolphin Network and 44 to 2 for the Prisoners network).

The Figure 4.5 compares the positions of the key nodes identified by the Eigenvector centrality approach and the positions of the key nodes identified by the multi-objective approach. Clearly, the key nodes identified by the multi-objective approach are well spread throughout the network.

When all the objectives in $\mathcal{M}$ are considered in a single step to identify the sets of key players using multi-objective optimization, the number of non-dominated solutions identified is large. For example, when all the objectives were considered in a single step, the NSGA-II algorithm identifies 549 sets of non-dominated key players for the Dolphin network and 249 sets of non-dominated key players for the Prisoners network. Recall that the two step process described above identified only 3 sets of key players for the networks

Table 4.6: Set of Key Players found for Dolphin Network from Pareto front and respective centrality values

Set ID	Set of Key Players					DC	BC	CC	PR
1	26	20	45	14	13	0.31	0.38	0.49	0.052
2	56	49	58	11	60	0.33	0.39	0.52	0.053
3	**56**	**2**	**60**	**45**	**14**	**0.39**	**0.49**	**0.59**	**0.065**
4	56	49	58	60	55	0.28	0.35	0.46	0.047
			⋮			⋮	⋮	⋮	⋮
22	**56**	**14**	**60**	**45**	**46**	**0.42**	**0.46**	**0.6**	**0.068**
			⋮			⋮	⋮	⋮	⋮
30	**9**	**20**	**45**	**14**	**17**	**0.40**	**0.47**	**0.58**	**0.065**
			⋮			⋮	⋮	⋮	⋮

Table 4.7: Set of Key Players found for Prisoners Network from Pareto front and respective centrality values

Set ID	Set of Key Players					DC	BC	CC	PR
1	0	10	12	18	34	0.13	0.11	0.39	0.14
2	0	10	13	18	34	0.13	0.12	0.38	0.09
3	0	10	13	18	51	0.26	0.31	0.49	0.44
4	0	10	18	21	37	0.13	0.10	0.36	0.13
			⋮			⋮	⋮	⋮	⋮
31	**7**	**36**	**40**	**55**	**15**	**0.47**	**0.70**	**0.60**	**0.59**
32	**7**	**36**	**40**	**54**	**15**	**0.48**	**0.66**	**0.60**	**0.59**
			⋮			⋮	⋮	⋮	⋮

Note : DC, BC, CC and PR stand for Degree centrality, Betweenness centrality, Closeness centrality and PageRank respectively

Dolphin network and 2 sets of key players for the Prisoners network. Since the users would be more interested in identifying a small number of sets of key players for their applications, the two step process is more useful.

This approach not only allows us to identify sets of key players that optimizes both objectives, but also allows us to evaluate other key player identification algorithms with regard to selected objectives. Once the non-dominated set is obtained using the first two selected objectives, the sets of key players identified by previously proposed methods can be compared against the obtained non-dominated set[3]. For example, consider the address-

[3]Key player identification methods of Degree centrality, Betweenness centrality [81], Eigenvector centrality [13], Pagerank[94], Borgatti's KPP Positive[14] , Borgatti's KPP Negative[14], Principal Component

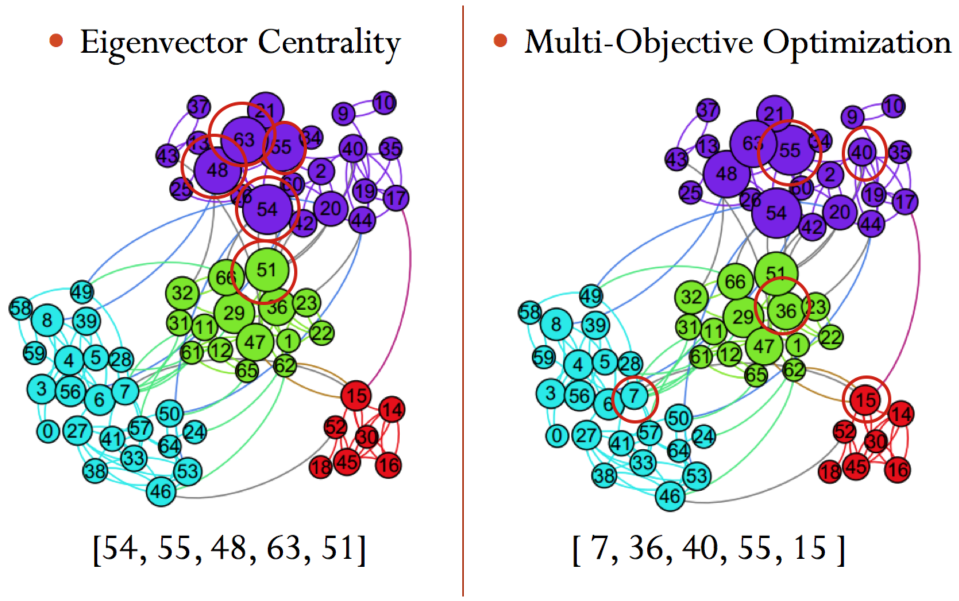


Fig. 4.5: Comparison of the positions of the key players identified by the Eigenvector centrality approach vs the positions of the key players identified by multi-objective approach

ing of the deficiency of Eigenvector centrality again. Each of the above methods assigns a certain value for average Eigenvector centrality and another value for distance between the key players. The points corresponding to these two values are shown in Figures 4.3(a) and 4.3(b).

All the sets of key players identified by previously proposed algorithms are dominated by the solutions in the non-dominated set in Figures 4.3(a) and 4.3(b). According to Figure 4.3(a):

- The sets of key players identified by all the methods mentioned above are dominated by the solutions in the Pareto front.

- The key player set found by Principal Component centrality is near the Pareto front (Euclidean distance of 0.44 to the closest point in the Pareto front) while the key player set found by Betweenness centrality is far from the Pareto front (Euclidean distance of 1.04 to the closest point in the Pareto front). This indicates that Princi-

centrality[60], KPP Positive using Information theory [93], KPP Negative using Information theory [93], K-shell [68] are compared here.

pal Component centrality can identify key player sets better suited to optimize both Eigenvector centrality and distance between key players than Betweenness centrality.

Given a Pareto optimal set of solutions, each of the methods discussed in Section 3.3.1 and the *Leave-k-out* approach select different solutions from the Pareto set. As the solutions in a Pareto set are non-dominating with respect to each other, there is no obvious way of defining metrics to evaluate the solutions in the Pareto set with each other. Hence, we evaluate the *Leave-k-out* approach against the other methods of selecting solutions in terms of the *Eventual Information Limitation (EIL) problem* (in Section 4.5.1), and the *Immunization problem* in Section 5.3.2.

4.5 Applications of multi-objective k-key players

4.5.1 Eventual Information Limitation problem

This section is focused on evaluating the relative quality of the key player sets identified by the multi-objective approach discussed in Sections 4.2 and 4.4.

Online social networks offer an excellent platform for information dissemination. An individual (X) in a social network is connected to his/her peers through social links, and gets exposed to the pieces of information shared by his/her peers. If X decides to accept a certain piece of information, X can also share this information with X's peers. This sharing and acceptance process happens iteratively and leads to the diffusion of information through the social network. Multiple computational models have been proposed over the years to capture the dynamics of such information spread in social networks.

In the real world, multiple pieces of such information can get diffused simultaneously through the network of interest. If these multiple pieces of information are related (co-operative or competitive), they do not spread independently [87]. Marketing campaigns launched by a certain company for two of its products can be considered as an example for cooperative diffusion of information. If one of these products gets endorsed by many individuals in a population, this in turn could improve the 'word of mouth' spread of popularity of the other product as well, since both the products belong to the same company. Thus, the spread of information is inter-dependent and the two contagions cooperate as they mutually help each other in spreading through the network. Marketing campaigns launched by two different companies for competitive products provide an example for competitive contagions. In this case, the individuals who accept a certain product A in the competition are less likely to endorse the other product B in the competition. This decreases the spread of popularity of product B in the social network. As is clear through the aforementioned examples, the models proposed for spread of independent information will not hold in the cases where inter-dependent information flows are spreading simultaneously.

When multiple campaigns cascade in social networks, the optimization problem can be formalized either as influence maximization or influence blocking maximization (or influence minimization) problem [114]. Assume the two campaigns cascading in the network are A and B, and the budget of each campaign is p, where p is the maximum number of nodes selected by a campaign as its initial set of seeds. The problem of interest here is to pick an initial set of seeds for each campaign to maximize the number of nodes recruited for the same campaign at the time the campaigns stabilize. The campaigns stabilize when no more nodes can be recruited by any of the two campaigns. When the problem is formalized as an influence maximization problem the goal of cascade A is to recruit the initial set of nodes that has the capability of maximizing the number of recruited nodes for cascade A at the time the campaigns stabilize. Identification of the optimal set of nodes for this problem has been found to be NP-hard. But there is a greedy algorithm that yields a solution that is within 1-$\frac{1}{e}$ of the optimal solution [67, 17]. Since the greedy solution also tends to be computationally complex, most of the recent approaches to solve the problem propose heuristic measures to select the starting seeds for the desired optimization problem of interest. Network centrality measures [96, 45], which are commonly used for identifying key players in social networks are among such measures used as heuristics [17, 90].

In a social network, each node represents an individual, and an edge $e_{u,v}$ represents the influence from node u to v. Initially all nodes are inactive and the information diffusion process starts when a set of nodes (initial seeds) becomes activated in a certain campaign and starts to diffuse information to their neighbors. A node becomes active in a campaign depending on the probabilistic model involved in the diffusion process. In the classical independent cascade model, each activated node u has a single chance to activate its inactive neighbor v with a success probability of $p_{u,v}$, which is a predefined parameter.

Modeling multiple simultaneous cascades

To model the diffusion of multiple cascades evolving simultaneously in a network, several models have been proposed in recent years. Budak et al. [17] proposed two models called Multi-Campaign Independent Cascade Model (MCICM) and Campaign-Oblivious Independent Cascade Model (COICM).

To explain the MCICM model, consider a network $G = (V, E)$ consisting of $|V|$ nodes and $|E|$ edges. Consider a situation where two cascades (R and L) evolving simultaneously in a network. The cascade R is used to denote a rumor and the cascade L is used to denote the limiting campaign which is used to limit the spread of cascade R. At any given time, any node v can be in one of 3 states; recruited by campaign R, recruited by campaign L or neutral. Two weights $p_{v,w}(R)$ and $p_{v,w}(L)$ are assigned to each edge $e_{v,w}$ which is used to model the direct influence the node v has on node w for cascade R and cascade L respectively.

The initial set of active nodes for cascade R (rumor creators) is denoted by A_R. When a node v first becomes active in campaign R in time t, it has a single chance to activate each of its neighbors w in campaign R and it succeeds with probability $p_{v,w}(R)$. The cascade R starts spreading through the network at $t = 0$ and the cascade L starts spreading after a delay d. The delay d is determined by the time taken to detect the spreading of the rumor R

in the network[4]. After the rumor R is identified, the campaign L starts to spread, after some delay (r) and attempts to send “good" information to the network. The campaign L has a budget b to recruit nodes, and suppose that to recruit a neutral node the campaign L should spend a unit cost. The initial set of nodes recruited by campaign L is denoted by A_L. After a node v becomes active in campaign L it can activate each of its neighbors w in campaign L and it succeeds with probability $p_{v,w}(L)$. The only difference in the COICM models is that the probability of each edge being live is independent of the campaign. In this setting only one probability $p_{v,w}$ is associated with each edge $e_{v,w}$. No matter which information reaches a node v, v forwards this information to its neighbor w with probability $p_{v,w}$. This model is proposed to simulate competing campaigns where the two information cascades are more likely to be of similar ‘quality’ and the nodes would agree to the campaign that reaches out to them first.

The problem addressed here is, *given a budget b, select a set of nodes A_L to serve as seeds for initial activation for the limiting campaign L, such that the number of nodes that adopt campaign L when the model stabilizes ($\pi(A_L)$) is maximized.* This problem is also known as the Eventual Influence Limitation (*EIL*) Problem.

As proved in [17], *EIL* is a NP-hard problem, so it is believed to be impossible to find the optimal set of key players A_L in polynomial time. Different algorithms have been used to select the set A_L and the results showed that in many cases, the quality of results obtained by heuristics such as degree centrality (for solving *EIL*), is comparable to computationally costly algorithms. We have evaluated the performance of different key player identification algorithms discussed in Section 2.1, and the multi-objective approach discussed earlier with regard to the *EIL* problem. A set of 5 key players ($b = 5$) selected by each key player identification algorithm was used as the seed for initial activation for L and eventually the $\pi(A_L)$ values generated by each set of key players were compared against each other.

[4]How to identify the spreading of a rumor is out of scope for this study

Application of the Leave-k-out approach for multi-objective optimization

The objectives that a set of key nodes should maximize for the *EIL* problem are assumed to be *Degree centrality*, *Betweenness centrality*, *Closeness centrality*, *Eigenvector centrality* and *PageRank*. The implications of these centrality measures are shown in Table 4.5.

When all 5 of the aforementioned objectives were optimized to identify 5 key nodes on Dolphin and Prisoners networks, the 5 objective optimization results in 67 and 72 solutions for the two networks respectively. In Section 3.3.1, we discussed the approaches that have been proposed to select solutions from the Pareto set and in Section 4.4.1 we introduced a novel approach (*Leave-k-out* approach) to select solutions from the Pareto set. In this section we evaluate the performance of the *Leave-k-out* approach.

1. **Performance of the selected solutions:**

   The aim of the *EIL* problem is to 'save' as many nodes as possible from receiving the gossip. Hence, with respect to the *EIL* problem, we measure the quality of the results by the number of nodes recruited by the limiting campaign when the cascades settle ($\pi(A_L)$). If $\pi(A_L)$ is higher, the set of initial seeds picked by the key node identification algorithm is better. Table 4.8 shows the comparison of the Pareto set pruning approaches on the number of nodes recruited by the limiting campaign. When multiple sets of key players were identified by a Pareto set pruning approach, the average performance of the identified sets was used for comparison against the other algorithms. Also, selecting k objectives to leave out for the *Leave-k-out* approach can be done in multiple ways, the results show for each case of *Leave-k-out*, are the averages of all cases. Each number shown in Table 4.8 is the average over 30 trials.

   According to the results shown in Section 4.5.1, *Leave-k-out* with $(k = 1)$ (or *Leave-one-out*) approach yields the slightly better performance when compared to other approaches of selecting solutions from the Pareto set. But, according to the results,

Table 4.8: Comparison of Pareto set pruning approaches on the number of nodes recruited by the limiting campaign

Method	Network	
	Dolphin	Prisoners
Leave-k-out (k=1)	44.39	42.14
Leave-k-out (k=2)	40.21	40.54
Leave-k-out (k=3)	38.92	38.88
Weighted sum	41.12	40.04
Average Ranking	42.47	39.98
Maximum Rank	38.97	37.84
Cluster centers	39.02	39.54
Ideal points	40.93	40.92
Angle based pruning	39.89	38.17
Favor Relation	39.47	39.16
K-optimality	41.97	40.42

no method significantly outperforms other methods. Hence, we conclude that when evaluated based on the performance on the *EIL* problem, all the considered methods for selecting best solutions from the Pareto set perform equally well.

2. **Comparison of Running times:**

   Next, we compare the running time of each algorithm proposed to select solutions from the Pareto set. The running time of each approach (in seconds) is shown in Table 4.9. According to the results, the *Leave-k-out* approach outperforms all other approaches significantly $(p < 0.001)$. The reduced running time of the *Leave-k-out* approach occurs due to the reduction of objectives in the multi-objective optimization. In the above experiment, *Leave-k-out (k=1)*, *Leave-k-out (k=2)*, and *Leave-k-out (k=3)* requires 4, 3 and 2 objective optimization in the first step respectively. All the other approaches require 5 objective optimization.

   The reason for reduced running time, when the number of objectives is reduced is as follows. We use two termination criteria for *NSGA-II* algorithm:

   i. Algorithm reaching the maximum number of generations allowed.

   ii. All of the objectives remain unimproved for a k number of consecutive genera-

Table 4.9: The comparison of running times (in seconds) of Pareto set pruning approaches

Method	Network	
	Dolphin	Prisoners
Leave-k-out (k=1)	74.45	73.82
Leave-k-out (k=2)	44.76	40.83
Leave-k-out (k=3)	31.83	30.21
Weighted sum	173.98	168.51
Average Ranking	175.22	170.32
Maximum Rank	183.23	182.53
Cluster centers	199.23	200.16
Ideal points	204.22	203.21
Angle based pruning	211.93	221.37
Favor Relation	821.92	711.49
K-optimality	378.34	311.54

tions ($k = 50$ is used in our experiments).

When the number of objectives in the *NSGA-II* algorithm is reduced, the algorithm terminates early, as the number of generations needed to reach the best fitness value (criteria ii) is reduced. The reduced number of generations needed for termination ensures that the number of function evaluations is less. The following are the numbers of function evaluations for different cases in the *EIL* problem. In each calculation, the number of individuals per generation is 50.

i. <u>5 objective optimization</u>

$$\begin{aligned} Average\ number\ of\ generations &= 731.6 \\ Average\ number\ of\ function\ evaluations &= 731.6 \times 50 \times 5 \\ &= 182900 \end{aligned}$$

ii. 4 objective optimization (Leave-k-out (k=1))

$$\begin{aligned} Average\ number\ of\ generations &= 418.83 \\ Average\ number\ of\ function\ evaluations &= 418.83 \times 50 \times 4 \\ &= 83766 \end{aligned}$$

iii. 3 objective optimization (Leave-k-out (k=2))

$$\begin{aligned} Average\ number\ of\ generations &= 268.17 \\ Average\ number\ of\ function\ evaluations &= 268.17 \times 50 \times 3 \\ &= 40225.5 \end{aligned}$$

iv. 2 objective optimization (Leave-k-out (k=3))

$$\begin{aligned} Average\ number\ of\ generations &= 182.47 \\ Average\ number\ of\ function\ evaluations &= 182.47 \times 50 \times 2 \\ &= 18247 \end{aligned}$$

3. **Quality of the solutions identified:**

To evaluate the quality of the solutions obtained by the *Leave-k-out* approach, we consider the probability that the solutions identified by the *Leave-k-out* approach are also solutions that belong to the Pareto front of the original multi-objective optimization problem. If this probability is greater, then the solutions identified by the *Leave-k-out* approach are closer to the Pareto front, and can be considered to be better. Table 4.10 shows the results. All the results shown are averages over 30 trials.

Leave-k-out (k=1) needs a 4 objective optimization and the evaluation of the objective that was left out. According to the results, the solutions obtained by *Leave-k-out (k=1)* were always found in the Pareto front of the 5 objective optimization of the *EIL* problem. Similarly, the solutions obtained by *Leave-k-out (k=2)*, and *Leave-*

Table 4.10: Average percentage of the solutions identified by the *Leave-k-out* approach are also solutions that belong to the Pareto front of the original multi-objective optimization of the EIL problem

	Dolphin	Prisoners
Leave-k-out (k=1)	1.0	1.0
Leave-k-out (k=2)	0.92	0.90
Leave-k-out (k=3)	0.81	0.81

k-out (k=3) have respective probabilities 0.92 and 0.81 of being in the Pareto front of the 5 objective optimization for the Dolphin network. This indicates that when more objectives are left out from the initial optimization, the quality of the solutions decreases. This also explains the reduction of performance among the solutions of *Leave-k-out*, when k increases (Table 4.8). A similar pattern is observed for the Prisoners network as well.

It is clear from the above results that the number of function evaluations decreases significantly when the number of objectives in the multi-objective optimization is decreased. Hence, the running time of the *Leave-k-out* method for selecting solutions is significantly less compared to other approaches. Similar results were obtained when the *NSGA-II* algorithm was replaced by the *SPEA2* multi-objective optimization algorithm.

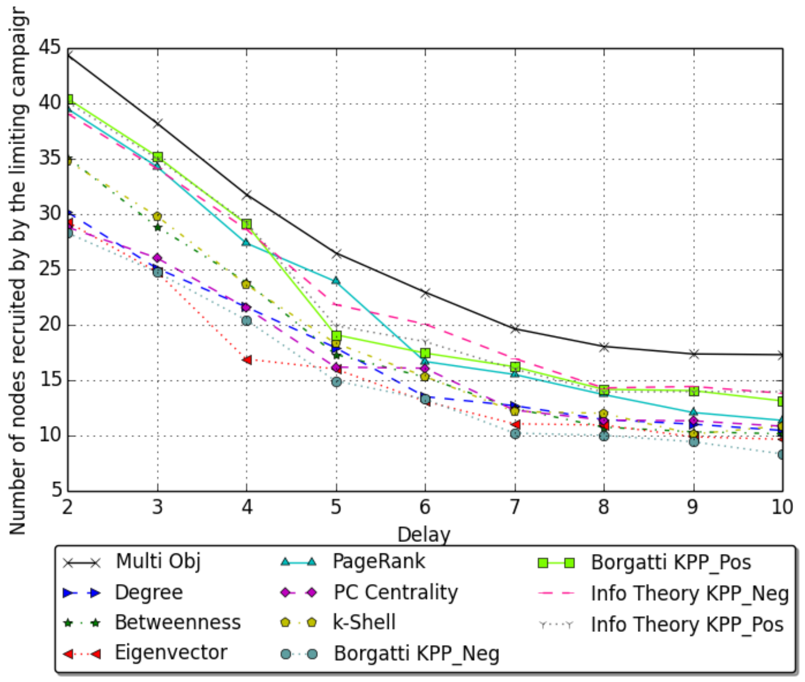
Selecting the appropriate k value for the *Leave-k-out* approach is a trade-off between performance and computational cost. For example, for the *EIL* problem, when $k = 1$, *Leave-k-out* shows the best performance (in Table 4.8), but requires the highest computational time (in Table 4.9) among all the *Leave-k-out* variants. In the same example, when $k = 3$, *Leave-k-out* shows the worst performance (in Table 4.8) among all the *Leave-k-out* approaches, but best computational time (in Table 4.9). Hence, selecting the appropriate k value depends on the application and the decision maker. For example, for the Dolphin network, selecting $k = 1$ over $k = 2$, delivers only a 10% increase in performance while increasing the computational time by 67%. Hence, for this application, the *Leave-k-out* with $k = 2$ is desirable.

Performance comparison of key node identification approaches on the EIL problem

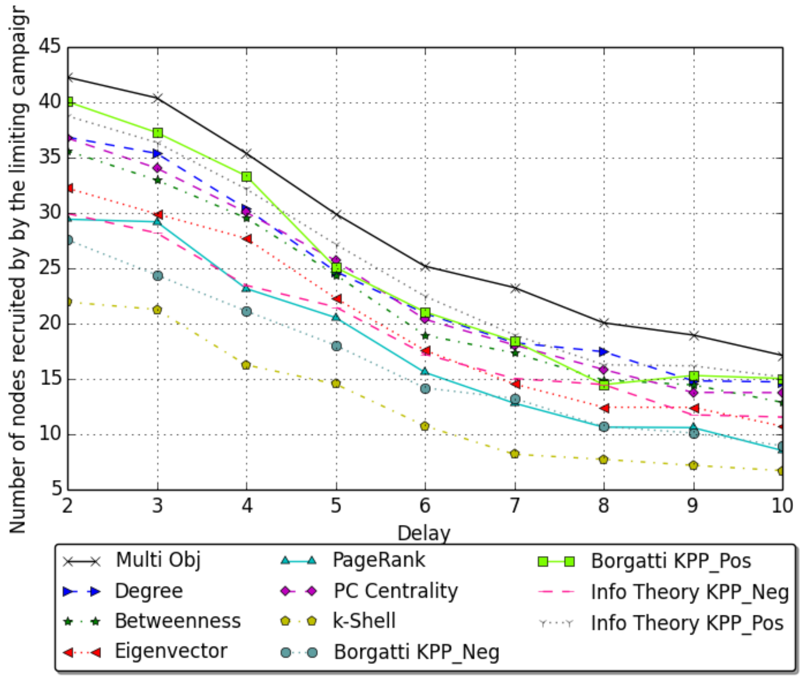
When comparing the performance of the multi-objective key nodes against other key node identification methods, we use the *Leave-k-out* approach with ($k = 2$) as our selection method. Figure 4.6(a) shows the variation of $\pi(A_L)$ for different key player identification algorithms with the delay r for the Dolphin network. Since the aim is to "save" as many nodes as possible from getting the gossip, the idea is to achieve high $\pi(A_L)$ values. As the delay increases, the number of nodes recruited by the gossip campaign is high, so the $\pi(A_L)$ value of each key player set decreases as expected. According to Figure 4.6(a), the $\pi(A_L)$ value of the key player set identified by the multi-objective approach is higher than the values obtained by all the other algorithms used in the comparison. Figure 4.6(b) shows the same plot with regard to Prisoners network and this figure also shows that the key player set identified by the multi-objective approach achieves high $\pi(A_L)$ compared to alternative key player identification methods. For both the networks, the number of nodes saved when the multi-objective approach was used to identify initial seeds (key nodes) is significantly high ($p < 0.001$) compared to previous approaches of selecting key nodes.

4.5.2 Improving the fault tolerance of the smart grid

The smart grid interconnects a power grid (network) and a communication network, and enables bidirectional flow of electricity and information. To prevent the cascading failures which occur when the disruptions in one network cause disruptions in the other network, robustness should be enhanced by increasing the number of links (edges) between the power grid and the information flow network. Given a budget which constrains the number of new links that can be added to 'strengthen' the network, the best strategy to determine where to add those new links remains an open research problem. We used the multi-objective key player identification approach to identify the best subset of nodes in the power network where new links can be added to improve the overall robustness of the smart grid, when constrained by resource limitations.



(a) Dolphin Network - Delay vs Number of nodes recruited by the Limiting Campaign



(b) Prison Network - Delay vs Number of nodes recruited by the Limiting Campaign

Fig. 4.6: Number of nodes recruited by the Limiting Campaign starting at differnt delays

The power network consists of entities needed for the generation and distribution of electricity, such as generating stations, sub-stations, and distribution stations, connected through the transmission and distribution power lines. The proper operation of the components of the power network requires continuous monitoring and control, performed by nodes in the communication network. Conversely, the communication network is also dependent on the proper operation of the power network, since each entity in the communication network needs to obtain power from an entity in the power network.

An important problem with such inter-dependent networks is that failures in one network can lead to failures in the other network, hence the possibility of catastrophic cascading failures in the system. A failure in a power station could result in failures of some communication stations as they require electricity from the failed power station, then the failures in the communication stations might cause failures in other power stations which receive control signals from the failed communication stations.

Such cascading failures can lead to blackouts on an enormous scale. In August 2003, a power blackout occurred in the northeastern United States and parts of Canada, affecting over 55 million people. It was announced later that the outage started from a computer malfunction in Ohio and cascaded into a widespread power grid failure [3]. According to the US Department of Energy, power failures cost the nation about $150 billion each year [76]. This cost may increase substantially when large-scale power blackouts occur. In addition to natural causes, such disasters can be caused by cascading failures stemming from cyber-attacks launched by malicious agents. Cyber-attacks similar to ‘Stuxnet’ have been reported in entities related to the United States power grid [91], and the blackout in Turkey in March 2015 was believed to be caused by a cyber-attack [1]. Hence it is important to identify the vulnerabilities of smart grid systems to cyber-attacks, and to find methods to mitigate the effects of such attacks. Our approach addresses this problem *via* the identification of key players in the relevant networks.

In related work, interdependent networks were introduced and studied for the effect of cascading failures by Buldyrev et al. [18]. The failure of a few nodes in the communication network will affect nodes in the power network, which will further affect nodes in the communication network. In their study, the two networks were assumed to be of the same size, and one-to-one correspondence was assumed between the nodes which are joined by inter-network links. In [105] a similar inter-dependent network model with multiple support inter-network links, was studied.

Huang et al. [57] modeled the smart grid as two scale-free inter-dependent networks generated using the generalized Barabasi-Albert model [88], and studied the robustness in a phase of cascading failures following random node failures/attacks on the communication network. They observed that when the power law parameters decrease, the number of surviving nodes (after cascading failures stabilize) increases.

Ruj and Pal [99] show that the smart grid disintegrates faster during targeted attacks compared to random attacks. For a network of 10000 nodes in the communication network and 1000 nodes in the power network, compromising about 2.2% of the communication network nodes with targeted attacks can destroy the whole network. Erdos-Renyi random networks [51] were found to be more resilient than scale free networks, when subjected to targeted attacks.

Huang et al. [58] showed that robustness can be improved by increasing the number of communication nodes (k) used to control each power grid node. When $k = 1$, the smart grid fails even at 2% random node attacks, and when k is increased to 2, the smart grid can operate even at 10% random node failures. When k is further increased to $k = 15$, the smart grid remains functional even if 60% of nodes in the communication network are destroyed. However, the difference in robustness between $k = 15$ and $k = 10$ is relatively small, i.e., we have diminishing returns in robustness for extra cost.

Schneider et al. [103] improve robustness by partially decoupling the inter-dependent network using 'autonomous' nodes. An autonomous power station would have an alternative (backup) communication node (which has its own energy power supply) to obtain control signals when one communication node fails. They showed that the number of autonomous nodes required to make the system robust can be reduced by a factor of five (compared to random autonomous node selection), if they used degree [75] or betweenness centrality [15] to identify the nodes to be made autonomous. In [97], Reis et al. showed that if inter-network links are made by the network hubs, the inter connected networks are stable and robust.

Modeling the Smart Grid

We model the smart grid using two interconnected networks: the power network and the communication network, whose details are discussed below.

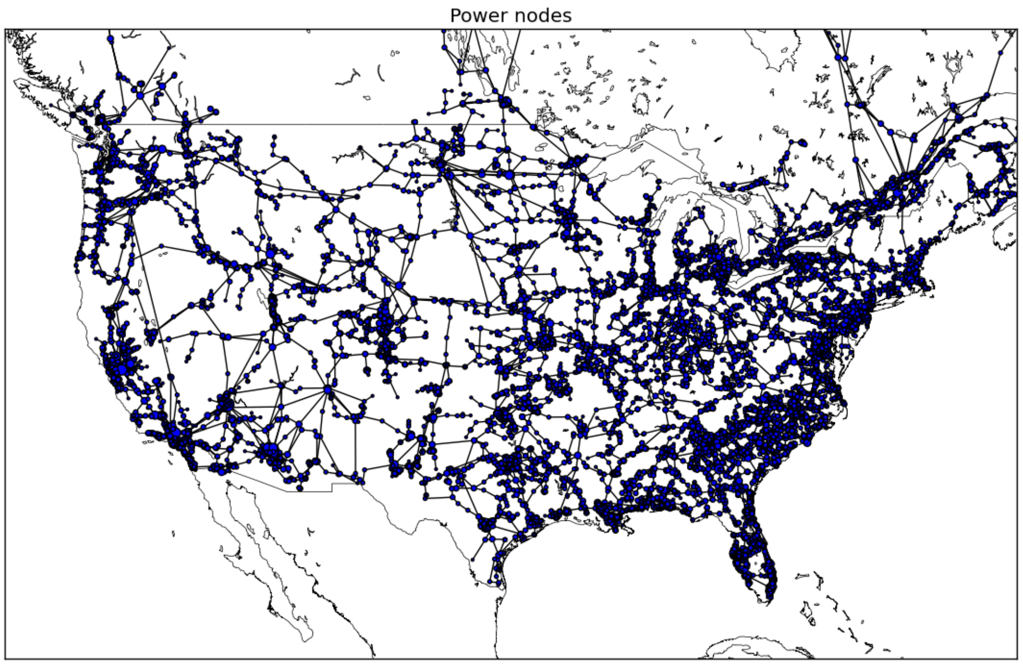
- *Power Network*: Here, the nodes represent generating stations, substations, transformers, etc., while edges represent high-voltage transmission and distribution lines between the nodes. The North-American high voltage power grid extracted by [119] and the degree distribution of the nodes in the power grid are shown in Figure 4.7. The network contains 14990 nodes and 18804 edges.

- *Communication Network*: This can be considered as a part of the internet, and the internet is known to be a scale free network [5].

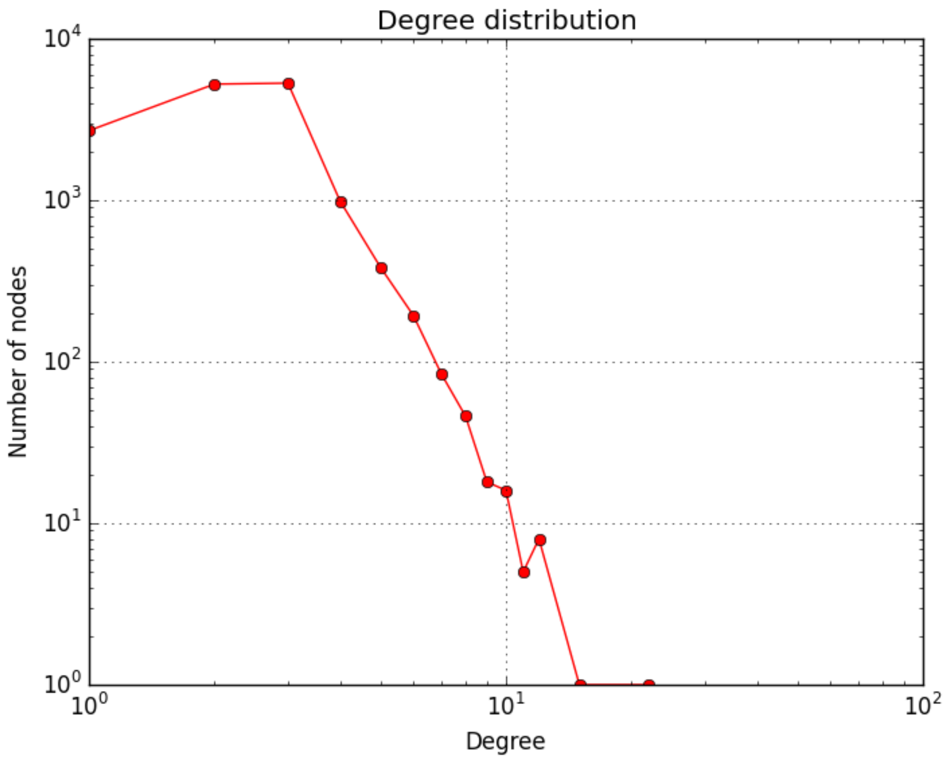
Since a real data set describing the inter-network connections is unavailable, we conduct simulations using the following assumptions:

1. Both the power network and the communication network are scale-free (SF) networks, where the degree distribution follows the power law, $p_k \propto k^{-\alpha}$, where p_k is the fraction of nodes with degree k and α is the power-law parameter specific to the network [57, 58, 99, 59].

2. A node is considered to be 'active' only if it belongs to the largest connected component in its own network, and has at least one inter-network link from an active node in the other network.



(a) North-American high-voltage power grid



(b) Degree distribution of the North-American high-voltage power grid

Fig. 4.7: North American power grid and the degree distribution

3. Inter-network connections could be many-to-many, with multiple inter-network links going to a node.

4. There will be more nodes in the communication network than the power network, since the same power node may supply power to more than one communication node.

We use two different models of the smart grid for our experiments.

Simulating the smart grid using a fully synthetic network

In this case, both the power network and the communication network are generated as separate scale free networks. Then, the inter-network edges are constructed as follows. Let N_p denote the power network, with n_p nodes and N_c denote the communication network with n_c nodes, where $n_c > n_p$. Let N_p^{in} and N_p^{out} denote the maximum in-degree and out-degree of a node in N_p, and let N_c^{in} and N_c^{out} denote the maximum in-degree and out-degree of a node in N_c. The following procedure is applied to add directed links from N_p to N_c. For each node $N_p^i \in N_p$,

i. Select a random number $N_p^i(out)$ in the range $[0 - N_p^{out}]$

ii. Let N_c' denote the set of nodes in N_c in which the in-degree is less than N_c^{in}. Select $N_p^i(out)$ nodes randomly from N_c'

iii. Add links from N_p^i to selected nodes of N_c.

Similarly directed links are added from N_c to N_p.

Simulating the smart grid using a semi-synthetic network

In this case, we use the data available in [119] to construct the power network, and use the location information of the power network to generate the communication network. We assume that the communication network has a geographical density similar to that of the power network. The communication network generation process is as follows.

i. Randomly select k nodes from the power grid.

ii. For each power grid node selected, randomly establish m communication network nodes within a neighborhood of a radius of r miles. The number of nodes in the communication network would be km.

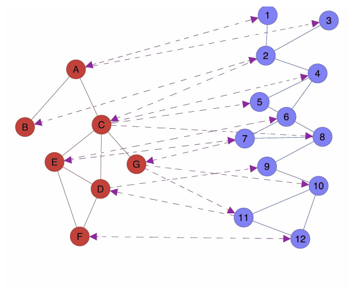
iii. Generate a power law sequence consistent with the size of the communication network.

iv. For each node in the communication network, assign a degree d from the generated power law sequence and connect to d randomly selected nodes in a neighborhood with a radius of r miles.

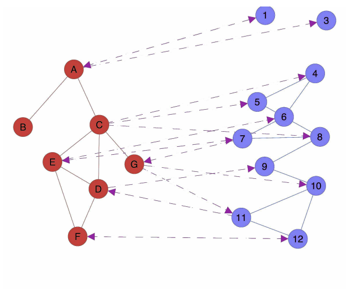
The process of inter-network edge creation is similar to the process presented in Section 4.5.2. But in this case, we connect inter-network nodes which are in a neighborhood with a radius of r miles.

Cascading failures in the smart grid

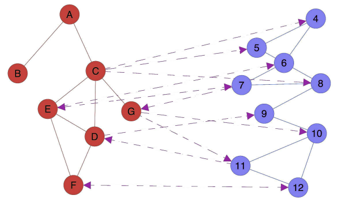
Figure 4.8 illustrates the occurrence of cascading failures in the smart grid. For this example, we assume that each power station $N_p^i \in N_P$ is controlled/monitored by one operation center (N_p^{in}), and provides power to up to three monitoring stations (N_p^{out}). Each monitoring/control station $N_c^j \in N_c$ can control up to two power stations (N_c^{out}), and receives power from one power station (N_c^{in}). The nodes denoted by blue color belong to the communication network and the nodes denoted by red color belong to the power grid. The cascade starts after the failure of node 2, and the Figures 4.8(a)-4.8(i) shows how failures cascade through both the networks. In these figures, the intra-network links are denoted by solid lines and the inter-network links are denoted by dashed lines. The power network nodes are labeled with numbers and the nodes in the communication network are labeled by English letters.



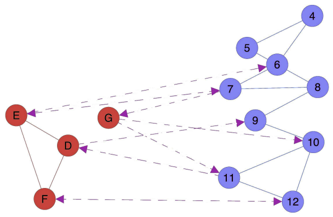
(a) State 0: Initial network. Node 2 will get attacked



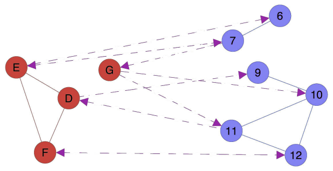
(b) State 1: Node 2 and its links removed



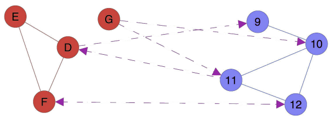
(c) State 2: Nodes 1 & 3 are not in the giant component of N_c; so get removed



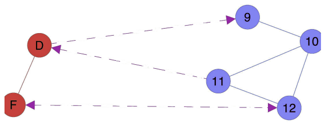
(d) State 3: Nodes A, B and C of N_p does not have any supporting links from N_c anymore; so get removed



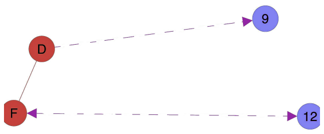
(e) State 4: Nodes 4, 5 and 8 lose their supporting links from N_p; so get removed



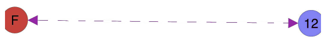
(f) State 5: Nodes 6 and 7 are not part of the giant component of the N_c anymore; so get removed



(g) State 6: Node E and G lose all supporting links from N_c; so get removed

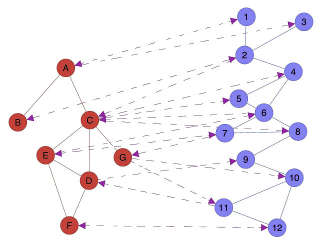


(h) State 7: Nodes 10 and 11 lose all supporting links from N_p; so get removed

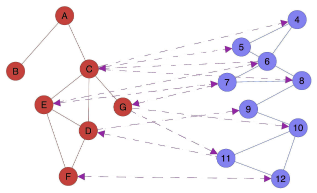


(i) Final State: Node D loses all supporting links from N_c, so will get removed. Then node 9 loses all supporting links from N_p; so get removed

Fig. 4.8: Cascading failures in a smart grid; the subgraph on the left (in red) shows the power network, and the network on the right (in blue) represents the communication network



(a) State 0: Initial network. Node 2 will get attacked



(b) Final State after cascades settle

Fig. 4.9: Result of cascading failures in smart grid when an extra controlling link is added to node C

Improving the robustness of the smart grid

Increasing the number of links between the two interrelated networks can increase the robustness of the smart grid for cascading failures [58]. The redundancy of a power station node N_p^a will be increased by adding extra links from the communication network to control that power station.

For example, node C in the power network in Figure 4.8 receives an extra control signal from node 6 in the communication network. Figure 4.9 depicts the initial network and the final state of the network after the cascading failures (as in Section 3). Thus, a single link addition increases the number of surviving active nodes in the power network from 14% to 71%.

The failure of a key node would cause more damage compared to the failure of a less significant node. For example, in the case of the power network, the failure of a larger power station would cause more power outages compared to a small scale power station. Thus, given a budget to add more links to the network, the robustness of the network could be improved by adding links targeted to increase the redundancy of links connected to key nodes in the network, rather than adding links randomly. In the example shown in Figure 4.9, node C was a highly connected node in the power network.

We now consider what strategy should be used to increase the robustness of the smart grid maximally, measured in terms of the number of nodes surviving after cascading node failures stabilize. The following link addition strategies are considered:

1. *Random* - In this case, the heads of the links are chosen randomly from the power network. For each of these chosen power network nodes, the tail of the link (source of the link) is selected at random from the nodes in the communication network.

2. *Degree Centrality* - Here, the heads of the links are chosen from the nodes with the highest degree in the power network. For each of the selected nodes, the tail of the link is selected randomly from the nodes in the communication network.

3. *Eigenvalue Centrality* - The heads of the links are chosen from the nodes with highest Eigenvector centrality in the power network.

4. *Betweenness Centrality* - The heads of the links are chosen from the nodes with highest Betweenness centrality in the power network.

5. *Multi-objective* - In this case, the heads of the links are chosen from the nodes in the multi-objective key player set in the power network, as described in Section 4.2. The rest of the edge creation process remains the same, as with the previous strategies.

In the multi-objective approach, we assume the set of key players should have high Degree centrality, Betweenness centrality, Eigenvector Centrality and Pagerank, and use those properties to optimize the multi-objective optimization process. For the experiments in Section 4.5.2, we selected key player sets of 10 nodes, which are non-dominated with respect to each other in the two selected objectives. To reduce the number of non-dominated key player sets identified, we represent the set of key nodes as a single *super node* and then evaluate the performance of this super node in terms of rest of the characteristics a set of key players should possess. In this case, we use *Leave-k-out* approach with $k = 2$.

Experimental Results

We assume that each power station $N_p^i \in N_P$ is controlled by one operation center, and provides power to upto three communication nodes (monitoring stations) each of which ($N_c^j \in N_c$) can control a power station, and receives power from a power station. Both the power network and the communication network are assumed to be scale-free, as in [57, 58, 99, 59].

Smart grid as a fully synthetic network

In this case, both the networks were constructed as scale free networks with power law parameter $\alpha = 2.5$. The power network contains 1000 nodes and the communication network contains 1500 nodes. Then, we increase the number of control links from the communication network to selected nodes in the power network. This increases the number of nodes surviving in the power network after the cascading failures settle. The robustness of the network was measured by the number of nodes surviving in the power network. Different key player selection algorithms were used to select the key nodes from the power network to add extra links, and were compared by examining the number of nodes surviving in the power network after the cascades settle.

1. Random Node Failures

   Table 4.11 shows the number of nodes surviving in the power network after the cascades occur due to random node failures. Here, the first column shows the number of extra links added to the smart grid. The second column shows the percentages of nodes attacked in the communication network and the third column lists the numbers of surviving nodes (after cascades) without adding any extra links to the network. Columns 4-7 correspond to the number of surviving nodes in the power network after extra links are added, based on Degree Centrality, Eigenvector Centrality and Betweenness Centrality, respectively. In each of these approaches, 10 key players

Table 4.11: Number of surviving nodes in the power network after a random node attack, in the fully synthetic model

% of links added	Nodes attacked	Nodes after attack	Links Added to 10 randomly selected nodes	Links added to top 10 Degree centrality key players	Links added to top 10 Eigenvector centrality key players	Links added to top 10 Betweenness centrality key players	Links added to top 10 multi-objective key players
1%	1%	252.24	261.42	318.60	318.67	345.66	**378.52**
	2%	248.87	257.61	314.74	314.42	341.92	**372.84**
	5%	238.74	245.68	304.66	303.23	328.54	**359.30**
	10%	221.74	226.47	285.18	283.79	305.61	**347.54**
	20%	185.60	187.96	246.24	243.01	260.06	**310.96**
5%	1%	252.24	269.20	339.44	343.40	345.07	**400.48**
	2%	248.87	264.48	336.46	340.41	341.95	**393.94**
	5%	238.74	252.00	326.72	329.49	332.95	**388.89**
	10%	221.74	231.28	310.43	312.63	314.59	**370.60**
	20%	185.60	190.33	276.83	274.01	279.80	**333.98**
10%	1%	252.24	286.04	354.77	355.98	355.83	**408.70**
	2%	248.87	281.78	350.15	351.94	351.08	**395.34**
	5%	238.74	270.09	339.67	337.01	341.92	**389.31**
	10%	221.74	246.44	321.18	323.14	323.01	**382.81**
	20%	185.60	202.51	283.93	282.89	286.42	**338.69**

Table 4.12: Standard deviation of number of nodes saved after random node attacks in the fully synthetic model

Random	Degree Centrality	Eigenvector Centrality	Betweenness Centrality	Multi-objective Key players
61.44	33.2	33.81	32.16	21.54

from the power network were selected to add extra links, and these extra links were distributed equally among these selected nodes. The last column shows the number of nodes saved in the power network by adding the links to the top 10 key players identified by the multi-objective key player identification algorithm described in Section 4.2. Results shown are averages over 100 different trials.

In all cases, adding extra links to the set of 10 key players identified by the multi-objective key player approach helps to save significantly more nodes in the power

Table 4.13: Number of surviving nodes in the power network after a targeted node attack in the fully synthetic model

% of links added	Nodes attacked	Nodes after attack	Links Added to 10 randomly selected nodes	Links added to top 10 Degree centrality key players	Links added to top 10 Eigenvector centrality key players	Links added to top 10 Betweenness centrality key players	Links added to top 10 multi-objective key players
1%	1%	234.18	243.12	307.30	318.52	336.44	**371.81**
	2%	211.83	239.57	304.67	317.47	335.35	**360.96**
	5%	176.59	191.47	263.04	264.31	296.04	**346.30**
	10%	122.35	135.76	216.43	232.11	247.05	**290.39**
	20%	87.58	100.34	146.56	149.50	157.14	**201.96**
5%	1%	234.18	251.16	311.59	314.27	321.13	**357.24**
	2%	211.83	244.37	306.17	309.71	311.07	**344.22**
	5%	176.59	205.55	279.38	270.19	280.99	**329.52**
	10%	122.35	141.81	244.81	240.13	243.77	**300.54**
	20%	87.58	106.51	187.94	177.43	189.31	**227.15**
10%	1%	234.18	258.92	319.33	324.88	331.51	**364.27**
	2%	211.83	253.61	305.27	300.43	314.28	**352.39**
	5%	176.59	221.53	285.46	280.59	300.51	**338.94**
	10%	122.35	150.64	255.55	267.75	281.33	**320.51**
	20%	87.58	129.40	201.11	214.94	229.72	**260.76**

Table 4.14: Standard deviation of number of nodes saved after targeted node attacks in the fully synthetic model

Random	Degree Centrality	Eigenvector Centrality	Betweenness Centrality	Multi-objective Key players
54.66	24.59	25.27	25.43	**17.69**

network compared to other key player selection methods ($p < 0.001$). Table 4.12 compares the standard deviations between various node selection algorithms across all cases shown in Table 4.11. The values in Table 4.12 show that the multi-objective key player method not only results in the best averages, but also the best standard deviation, compared to other key player selection algorithms.

2. Targeted Node Attacks

In targeted attacks, a node's probability of getting attacked is assumed to be proportional to the degree of the node. Table 4.13 shows the numbers of nodes saved in the power network after the failure cascades that happen due to targeted node failures. Each number in the table is the average over 100 scale free network simulation trials. Our results show that even in the case of targeted attacks, adding extra links to the set of 10 key players identified by the multi-objective key player approach helps to save significantly more nodes in the power network compared to other key player selection methods $(p < 0.001)$. According to the results in Table 4.14, the standard deviation of the multi-objective approach was lower than those of the other key player selection methods.

Smart grid as a semi-synthetic network

As mentioned in Section 4.5.2, in this case we used the real data for the power network and the generated data for the communication network. Table 4.15 shows the number of nodes surviving in the power network when the cascades settle, after different algorithms were used to add new edges to the network. The generated communication network depends on the set of k initial nodes that are identified from the power network, the set of m points randomly picked from the neighborhood of a radius of r miles and the generated power law sequence that we used to determine the degrees of the km nodes. The results presented here are averages over 30 semi-synthetic networks generated. In this study, the parameter values of $k = 4000$ and $m = 5$ were used. The results show that adding extra links to the set of 10 key players identified by the multi-objective key player approach helps to save significantly more nodes in the power network compared to other key player selection methods $(p < 0.001)$.

Table 4.15: Number of surviving nodes in the power network after a targeted node attack in the semi-synthetic model

% of links added	Nodes attacked	Nodes after attack	Links Added to 10 randomly selected nodes	Links added to top 10 Degree centrality key players	Links added to top 10 Eigenvector centrality key players	Links added to top 10 Betweenness centrality key players	Links added to top 10 multi-objective key players
1%	1%	6782.3	6928.8	7292.1	7222.6	7314.2	**7432.5**
	2%	6634.4	6804.2	7124.8	7198.5	7234.5	**7318.4**
	5%	5987.9	6219.2	6727.4	6628.2	7003.8	**7199.7**
	10%	4100.4	4245.3	5175.1	5119.3	5321.3	**5934.2**
	20%	472.9	667.3	2145.8	2284.9	2467.2	**3264.7**
5%	1%	6782.3	7284.3	7712.8	7841.9	7932.5	**8134.2**
	2%	6634.4	7024.2	7561.3	7624.7	7723.2	**8022.8**
	5%	5987.9	6418.3	7024.2	7285.6	7498.3	**7832.4**
	10%	4100.4	4468.6	5423.7	5715.1	6123.8	**6893.8**
	20%	472.9	841.7	2598.5	2728.7	3167.3	**3728.3**
10%	1%	6782.3	7422.1	7843.4	7717.7	8082.4	**8312.3**
	2%	6634.4	7187.9	7711.9	7698.8	7984.2	**8234.2**
	5%	5987.9	6600.2	7430.1	7583.5	7729.8	**8092.4**
	10%	4100.4	4672.2	6046.2	6317.1	6532.1	**7373.2**
	20%	472.9	1034.1	3198.5	3315.3	3782.9	**4573.6**

Table 4.16: Standard deviation of number of nodes saved after targeted node attacks in the semi-synthetic model

Random	Degree Centrality	Eigenvector Centrality	Betweenness Centrality	Multi-objective Key players
893.45	478.93	483.25	476.32	**321.74**

4.6 Concluding Remarks

Previously proposed approaches for key node identification have focused on one objective of interest. This leads to multiple deficiencies in the sets of key nodes identified. To alleviate these deficiencies, we proposed the multi-objective optimization approach for key node identification.

We have also proposed a novel algorithm (*Leave-k-out* approach) which obtains a small number of solutions for a multi-objective optimization problem, and reduces the computational cost significantly compared to previously proposed algorithms. We evaluate the *Leave-k-out* approach against the other methods of selecting solutions from a Pareto set in terms of the *Eventual Information Limitation (EIL) problem* (in Section 4.5.1). The experimental results show that the *Leave-k-out* approach performs as well as other methods of selecting solutions from the Pareto optimal set. When the running time is considered, the *Leave-k-out* approach outperforms all other approaches significantly. Hence we use the *Leave-k-out* approach to identify solutions for multi-objective optimization problems.

We show that key nodes identified using multi-objective optimization approach alleviate the aforementioned deficiencies. We use two well known applications, viz., *Eventual Information Limitation (EIL)* problem and improving the fault tolerance of the *Smart Grid*, to show that the key nodes identified using the multi-objective optimization approach outperform the previous approaches.

CHAPTER 5
REDUCING THE COMPUTATIONAL TIME

Many real life networks contain thousands of nodes and edges. As the size and the complexity of the networks increase, so do the running times of the network analysis measures. The application of the proposed multi-objective approach to key player identification depends on the computational complexity of individual network centrality measures and on the computational complexity of the optimization algorithm (such as NSGA-II). This chapter focuses on how network sampling can be used to reduce the running times without compromising much on the quality of key nodes obtained.

This chapter is organized as follows. The first section gives an overview of network sampling methods. The second section introduces the degree centrality based sampling approach that we propose to reduce the running time of the key node identification problem. The last section applies the of multi-objective key player sets obtained on degree centrality based sampled networks to address two well known problems. The results show that the multi-objective key player sets identified on sampled networks perform better than single objective key player sets identified by applying the algorithms on the entire network.

We use several datasets that are commonly used by network science researchers and are publicly available. The descriptions of the networks are shown in Table 5.1. Table 5.2 shows the running times of the centrality calculation algorithms for the datasets shown in Table 5.1.

Table 5.1: Statistics of the largest connected component in the networks and description

Network	Nodes	Edges	Description
ca-GrQc[1]	4158	13422	Scientific collaborations network between authors submitted to General Relativity and Quantum Cosmology category of e-print arXiv
PGP[2]	10680	24316	Interaction network of users of the Pretty Good Privacy (PGP) algorithm
ca-HepPh[3]	11204	117619	Scientific collaborations network between authors submitted to Astro Physics category of e-print arXiv
G-plus[4]	23628	39242	Contains Google+ (user to user) links. A node represents a user, and an edge denotes that one user has the other user in his circles.

Table 5.2: Running times for centrality calculations in seconds (Averages over 30 runs)

Network	Time for DC	Time for EC	Time for PR	Time for BC
ca-GrQc	0.008	0.88	0.62	87.49
PGP	0.011	2.15	1.67	707.2
ca-HepPh	0.021	2.16	5.66	1487.58
G-plus	0.038	4.71	4.34	4245.79

Note : DC, EC, PR and BC stand for Degree centrality, Eigenvector centrality, PageRank and Betweenness centrality respectively

5.1 Network Sampling

Network sampling has been used widely in social network literature in order to reduce the computational space and time cost to manageable limits. Different sampling techniques have been proposed, in order to obtain samples which are close to the original network

[1]Source: http://snap.stanford.edu/data/ca-GrQc.html
[2]Source: http://konect.uni-koblenz.de/networks/arenas-pgp
[3]Source: http://snap.stanford.edu/data/ca-HepPh.html
[4]Source: http://konect.uni-koblenz.de/networks/ego-gplus

in terms of the desired properties [66, 73, 123, 74]. Some of the most popular network sampling approaches are the following:

1. Random (RN): Nodes are picked uniformly at random from the network, and then the subgraph induced by the chosen nodes is selected. Such random sampling can result in disconnected networks.

2. Random walk (RW): This approach starts from a randomly selected node and selects one neighbor with an equal probability among all the links, and does the same from the newly reached node. This process continues until the desired number of nodes is reached.

3. Breadth First Search (BFS) or Snowball sampling: This approach starts with a randomly selected node and selects all the neighbors to the sample, and proceeds with the same process from the selected neighbors.

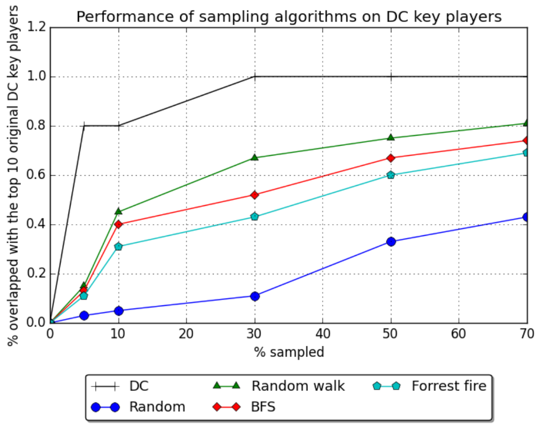
4. Forest Fire (FF): This approach is similar to BFS, but each neighbor of a selected node is only selected to the sample with a pre-defined probability p; a value of $p = 0.7$ is used in the experiments of this study [74].

A good sampling method for the key node identification problem should retain most of the key nodes of the original network in the sampled network. In the following section we propose a new sampling approach that perform better than existing sampling algorithms when applied to the key node identification problem.

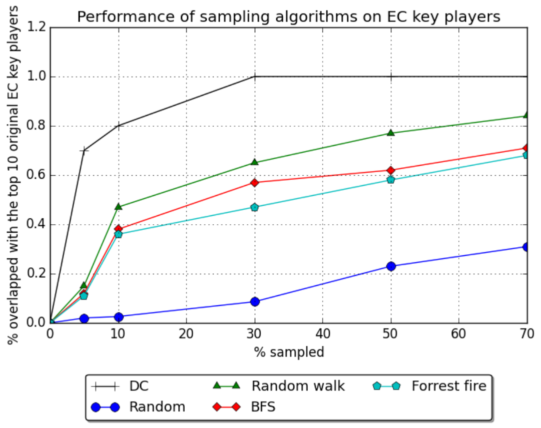
5.2 Degree centrality based sampling

Correlations among the network centrality measures have been studied in [72, 115]. The results suggest that degree centrality is highly correlated with other centrality measures. Since degree centrality is the least computationally expensive of the centrality measures to calculate, we investigated whether the key players with respect to other centrality measures also get retained in the sample, if the networks were sampled with degree centrality. To obtain an $\alpha\%$ sample of the network, we calculate the degree centrality of all the nodes of the original network and extract the network induced by the nodes with top $\alpha\%$ degree centrality.

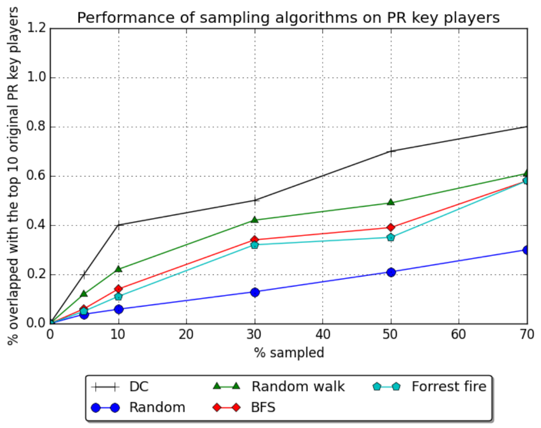
To evaluate the performance of different sampling methods we compare the number of key nodes of the original network retained in the sampled network.



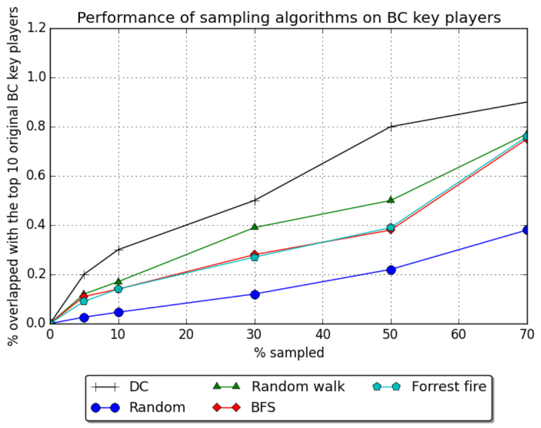
(a) Degree Centrality



(b) Eigenvector Centrality



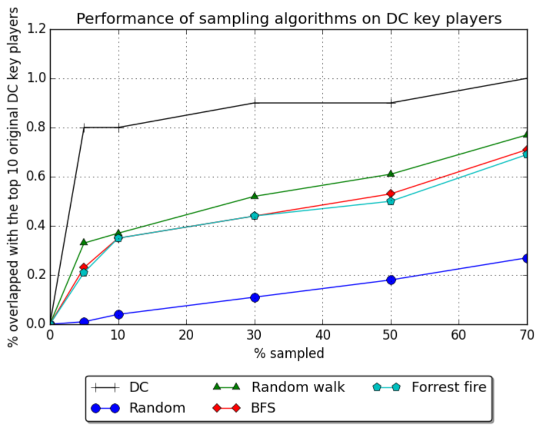
(c) PageRank



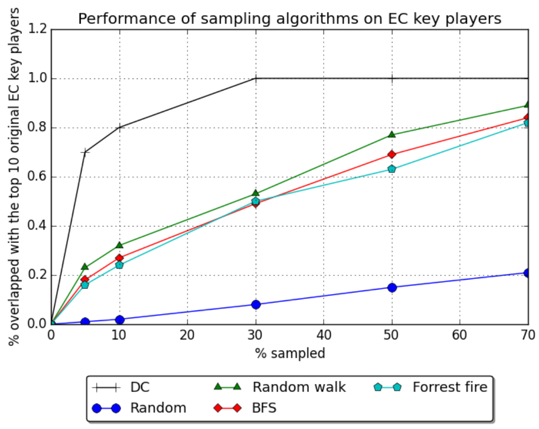
(d) Betweenness Centrality

Fig. 5.1: Comparison of sampling algorithms on the ca-GrQC network: performance in retaining the original network's top 10 key players using different centrality measures

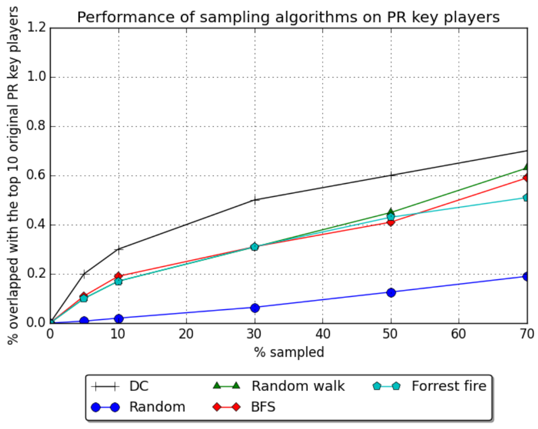
In Figures 5.1(a)-5.1(d), the Y axis represents the percentage of top 10 key nodes in the original network that overlap with those obtained using the sampled network, and the



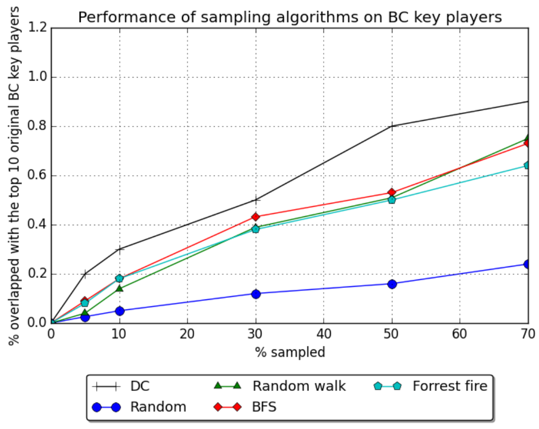
(a) Degree Centrality



(b) Eigenvector Centrality



(c) PageRank



(d) Betweenness Centrality

Fig. 5.2: Comparison of sampling algorithms on the PGP network: performance in retaining the original network's top 10 key players using different centrality measures

X axis represents the sampling percentage. In Figure 5.1(a) we compare the fraction of the top 10 key nodes identified using different sampling algorithms that overlap with the top 10 key nodes of the original network. Results show that the degree centrality based sampling method manages to retain most of the original network's key nodes in the sample compared to other sampling methods. A similar pattern is seen in Figures 5.1(b)-5.1(d), when the results of the sampling methods are compared against the number of key players obtained from Eigenvector centrality, PageRank and Betweenness centrality respectively. When 10 key nodes are identified on the 50% degree centrality sample, and compared against the 10 key nodes identified on the original network, we observe that the degree centrality based sample manages to retain all the DC based key players, all the EC based key players, 70% of the PR based key players and 80% of the BC based key players. A similar pattern of results is seen for the PGP network, as shown in Figure 5.2.

Hence, from the results we can conclude that degree centrality based sampling outperforms other sampling methods when the key nodes retained in the sampled networks are compared.

Next, in Tables 5.3-5.5 we compare the running times required for the top 10 key player identification when degree centrality based sampling is used. Each running time shown in the table is the average of centrality calculation over 30 trials. The running times of the centrality calculations in the original networks are shown in Table 5.2. According to the results, when 50% degree centrality based sampling is used, the running times for Eigenvector centrality reduce by 78%, 79%, 73% and 83% for ca-GrQc, PGP, ca-HepPh and G-plus networks respectively. For Betweenness centrality calculation, the running time reduces by 80%, 84%, 84% and 81% for ca-GrQc, PGP, ca-HepPh and G-plus networks respectively. Recall (from the results shown in Figure 5.1), that when the 10 key nodes are identified on the 50% degree centrality sample, and the 10 key nodes are identified on the original network are compared, the degree centrality based sample manages to retain all the DC based key players, all the EC based key players, 70% of the PR based key players

Table 5.3: Time taken to identify EC key players with degree centrality sampling (seconds)

Network	5% sample	10% sample	30% sample	50% sample	70% sample
ca-GrQc	0.01	0.01	0.04	0.19	0.32
PGP	0.03	0.05	0.10	0.45	0.39
ca-HepPh	0.17	0.45	0.51	0.57	0.76
G-plus	0.23	0.54	0.65	0.77	0.91

Table 5.4: Time taken to identify PR key players with degree centrality sampling (seconds)

Network	5% sample	10% sample	30% sample	50% sample	70% sample
ca-GrQc	0.14	0.17	0.35	0.47	0.55
PGP	0.26	0.38	0.78	0.84	1.31
ca-HepPh	0.59	0.89	1.81	2.85	3.71
G-plus	1.69	2.42	2.96	3.35	3.82

Table 5.5: Time taken to identify BC key players with degree centrality sampling (seconds)

Network	5% sample	10% sample	30% sample	50% sample	70% sample
ca-GrQc	0.15	0.64	6.62	16.63	33.26
PGP	1.69	6.16	36.67	107.86	215.21
ca-HepPh	9.47	31.78	124.72	230.67	312.43
G-plus	9.05	36.11	211.02	800.91	1344.21

and 80% of the BC based key players.

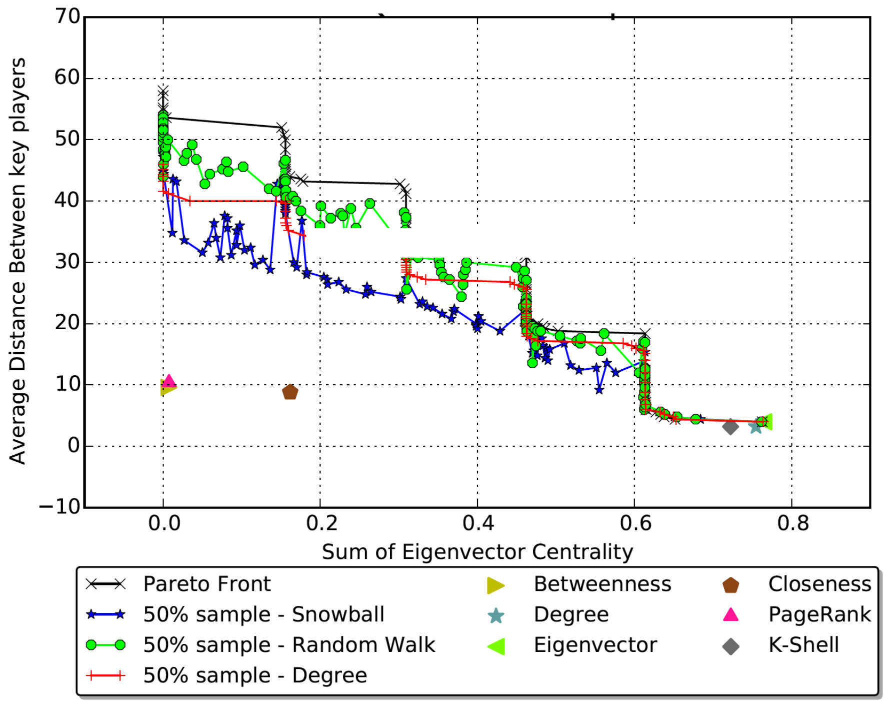
Hence we conclude that degree centrality based sampling is a successful technique that can be used to reduce the running times of centrality calculations in large networks. When the centrality values are calculated on the sampled network, the computational times are significantly less than the computational times of running the centrality algorithms on the original networks. In addition, even at 50% sampling level the set of key nodes identified on the sampled network highly overlaps with those identified on the original network.

5.2.1 Performance of degree centrality based sampling in key node identification

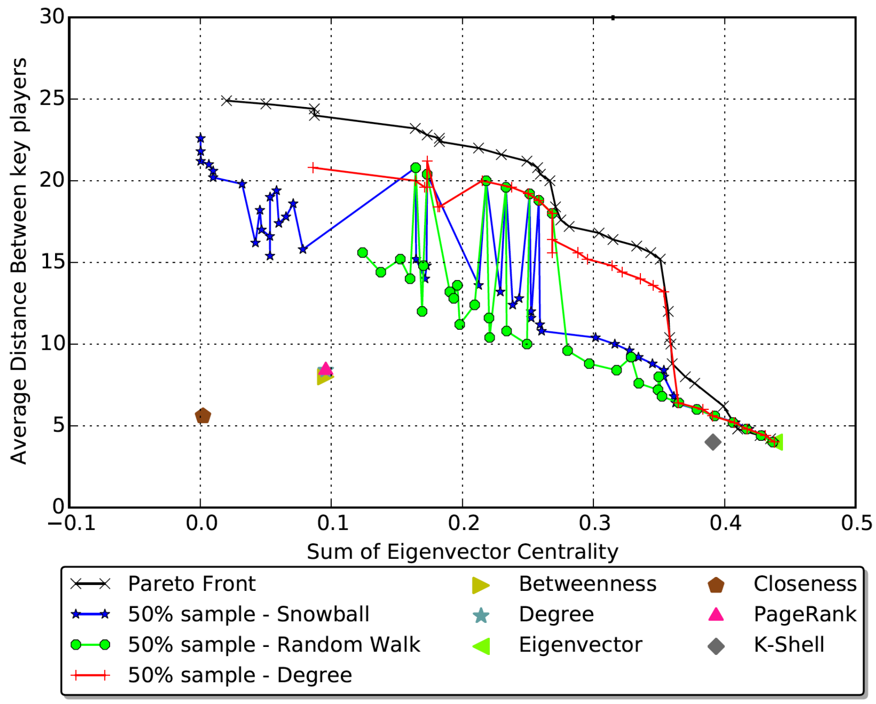
To evaluate the performance of sampling algorithms, we compare the Pareto fronts obtained by the key nodes identified by different sampling methods with the Pareto front obtained by the key players identified on the original network.

Figure 5.3 and Figure 5.4 show comparisons of the Pareto fronts obtained by the original network, sampled networks obtained by different sampling techniques and the corresponding points of the key player sets obtained by single objective centrality measures from the ca-GrQc and PGP networks. The Pareto fronts of the sampled networks have been obtained from the results of NSGA-II based evolutionary multi-objective optimization on 10 sampled networks and then the values for the two objectives obtained by the non-dominated key player sets on the original network are plotted in Figure 5.3 and 5.4. Although we considered 5 different types of network sampling methods including random walk, snowball, degree centrality, k-shell and multi-random walk, the results obtained from only random walk, snowball and degree centrality based sampling methods are shown in the figures to improve clarity. Note that the solutions in the Pareto fronts generated from the networks sampled by different sampling algorithms may not be non-dominated with respect to each other when the entire network is concerned.

According to the results shown by Figure 5.3 and 5.4 at sampling level of 50%, each sampling algorithm was able to capture sets of key players which achieve better values for both the objectives compared to single objective key player identification algorithms in both ca-GrQc and PGP networks. When different sampling algorithms are compared with each other, Degree centrality based sampling obtained sets of key players that are closest in the two objectives to the Pareto front obtained from the original network for both the networks considered here. It can also be seen that as the number of nodes retained by the sampling algorithms decreases, the performance of every sampling algorithm drops compared to the Pareto front of the original network. In particular, this drop is visible for smaller values

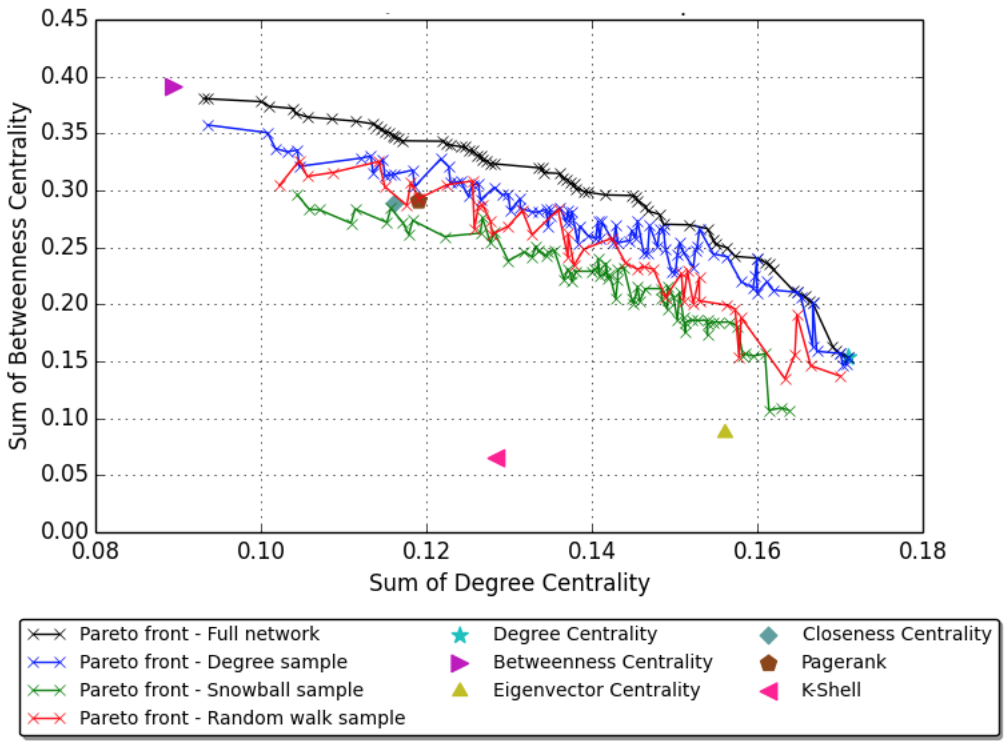


(a) ca-GrQc Network : Comparison of Pareto fronts of the original network, 50% sampled networks and single objective algorithms

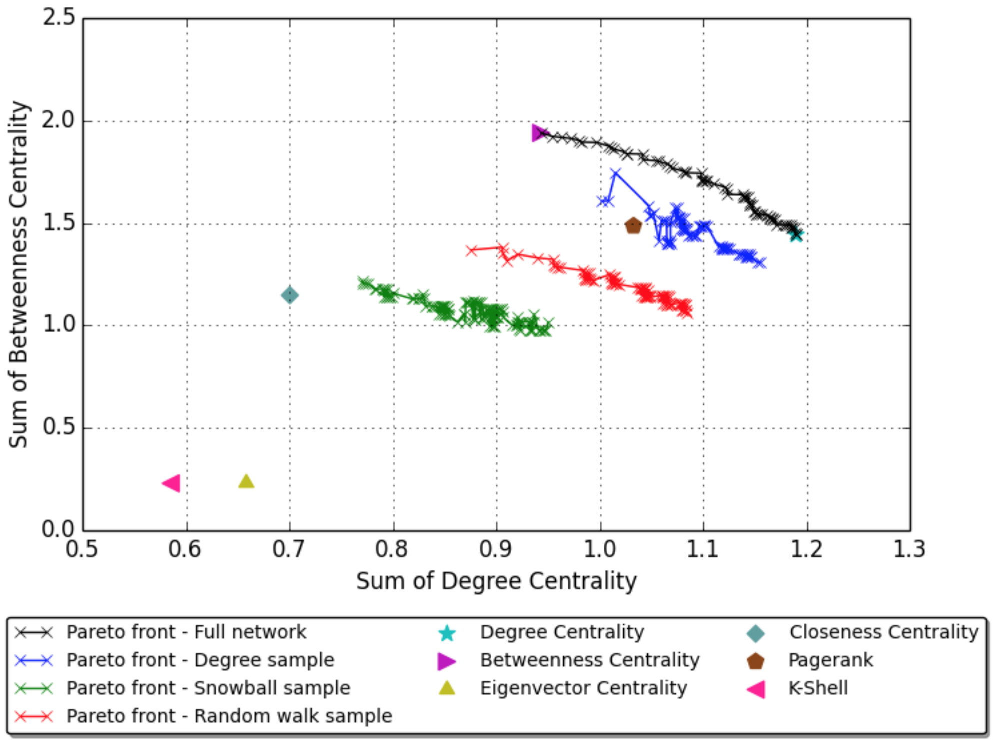


(b) PGP Network : Comparison of Pareto fronts of the original network, 50% sampled networks and single objective algorithms

Fig. 5.3: Pareto Fronts: Eigenvector centrality of the super node and Average distance between key players



(a) ca-GrQc Network : Comparison of Pareto fronts of the original network, 50% sampled networks and single objective algorithms



(b) PGP Network : Comparison of Pareto fronts of the original network, 50% sampled networks and single objective algorithms

Fig. 5.4: Pareto Fronts: Degree centrality of the super node and Betweenness centrality of the super node

Table 5.6: Improvements of running time using Degree centrality based sampling for multi-objective key player identification

Network	Original Network (mm:ss.ms)	30% Sample (mm:ss.ms)	50% Sample (mm:ss.ms)	70% Sample (mm:ss.ms)
ca-GrQc	4:58.8	2:13.8	3:15.2	3:42.0
PGP	8:55.4	4:09.8	6:26.2	7:31.9
ca-HepPh	11:16.0	5:00.8	7:32.7	8:27.5
G-plus	13:45.6	6:14.4	8:21.7	10:11.1

on the x-axis (Eigenvector centralities of the set of key players) and larger values on the y-axis (average distance between the set of key players). Since, we are only considering connected networks, as the number of nodes retained by the sampling algorithm decreases, the nodes of the sampled networks come from small portions of the original network. Thus, the high distance values between the key players decrease as the sampling become more aggressive.

Table 5.6 shows the improvement in running time when Degree centrality based sampling is used. Each running time is the average of executing NSGA-II based evolutionary multi-objective optimization 10 times. According to the results, at 50% Degree centrality based sampling, the running times reduce by 34%, 30%, 33% and 34% for ca-GrQc , PGP, ca-HepPh and G-plus networks respectively.

5.3 Applications of multi-objective key players identified on degree centrality sample

We use two well known problems to compare the performance of multi-objective key player sets obtained on a Degree centrality based sampled network and the key player sets obtained by single objective key player identification methods on the entire network. Single objective key player algorithms; DC (Degree centrality), EC (Eigenvector centrality), BC (Betweenness centrality), CC (Closeness centrality), PR (PageRank), PCC (Principal component centrality), ITP (Information theory based KPP POS), ITN (Information theory based KPP NEG) and KS (K shell) are compared in Tables 5.7-5.11 against the performance of the multi-objective approach. For both these problems we assume that the set of qualities key players should have will be measured by Degree Centrality, Betweenness Centrality, Closeness Centrality, Eigenvector Centrality and PageRank, and use *Leave-k-out* approach with ($k = 2$).

5.3.1 Performance of multi-objective key players in EIL problem

In this section, we use the EIL problem to compare the performance of multi-objective key player sets obtained on the 50% Degree centrality sample and the key player sets obtained by single objective key player identification methods on the entire network.

The effectiveness of the key player identification strategies was evaluated on the average number of nodes recruited by the Limiting Campaign at the time the model stabilizes. Table 5.7 shows the results after repeating the experiment for 1000 trials for the ca-GrQc network. The performance of the key player sets identified on the original network is better than the performance of the key player sets identified on the sampled network. Hence the performance of key player sets obtained on the sampled network by single objective key player identification methods is not shown in Table 5.7.

According to the results, it can be seen that multi objective approach with 50% sampling performs better than single objective key player approaches applied on the entire network ($p < 0.05$). In almost 90% of the cases the multi-objective approach outperforms all other approaches of key node identification.

Table 5.7: Average number of nodes recruited by the Limiting Campaign starting at different delays on ca-GrQc network

Key Players	Delay	DC	EC	BC	CC	PR	PCC	ITP	ITN	KS	MO
	4	1483.89	1450.76	1698.87	1621.50	1673.57	1543.28	1489.23	1469.16	1386.71	**1710.49**
	5	1334.76	1242.13	1513.00	1517.26	1469.08	1444.50	1382.37	1288.60	1161.79	**1622.33**
	6	1025.55	987.93	1162.43	1146.25	1061.51	1027.05	1035.76	1043.05	907.83	**1276.75**
5	7	604.89	564.87	764.96	723.64	774.02	555.23	625.53	648.22	445.79	**877.79**
	8	314.47	278.11	402.70	373.78	417.79	270.01	311.25	358.31	243.55	**449.63**
	9	150.81	147.56	226.96	173.04	235.32	126.41	129.55	227.21	108.20	**286.29**
	10	98.85	81.39	139.20	146.79	149.22	63.49	94.61	142.07	52.43	**177.95**
	4	1551.64	1466.49	**1826.44**	1641.15	1768.20	1599.99	1535.79	1520.62	1498.09	1796.32
	5	1409.33	1348.52	1553.06	1555.91	1544.47	1445.46	1387.98	1427.79	1324.38	**1699.12**
	6	1091.77	954.35	1262.89	1219.85	1255.58	1065.70	1049.47	1168.23	1061.16	**1427.21**
10	7	637.80	586.12	823.52	767.87	848.21	613.48	636.47	729.53	624.35	**957.88**
	8	365.97	282.87	533.75	464.74	502.48	340.21	352.21	490.72	333.17	**576.06**
	9	186.56	143.52	348.61	261.09	347.23	190.76	193.29	279.44	155.54	**380.88**
	10	132.93	81.67	226.63	172.26	212.27	94.44	108.37	188.73	92.27	**254.66**
	4	1555.39	1529.58	1869.67	1734.47	1838.92	1620.15	1510.02	1636.40	1498.91	**1897.47**
	5	1324.77	1381.32	1717.97	1580.24	1672.11	1335.23	1418.24	1418.80	1332.04	**1874.59**
	6	1098.99	1072.10	1401.54	1301.18	1286.84	1145.59	1069.58	1123.54	1023.53	**1511.97**
15	7	672.57	663.68	978.64	816.30	900.46	640.64	771.04	761.78	571.67	**1059.43**
	8	437.80	349.38	571.01	471.25	579.24	344.52	422.47	533.50	308.39	**673.29**
	9	238.13	191.21	388.32	302.49	**401.21**	189.87	204.05	308.86	176.54	399.93
	10	138.81	120.80	281.30	193.56	268.61	125.14	131.88	239.73	95.60	**321.17**
	4	1609.41	1560.58	1898.52	1800.86	1926.91	1683.75	1593.27	1675.51	1545.46	**2053.94**
	5	1450.98	1363.96	1741.68	1658.79	**1766.00**	1377.86	1462.17	1567.86	1351.01	1759.37
	6	1157.00	1077.10	1418.74	1304.44	1410.70	1106.98	1103.18	1237.77	1013.17	**1645.01**
20	7	702.43	674.63	999.33	898.37	1040.76	720.55	810.62	882.24	631.36	**1074.95**
	8	418.01	431.65	683.73	499.73	699.13	377.28	470.28	575.76	359.49	**889.51**
	9	240.02	205.69	414.40	359.06	458.87	235.64	277.10	357.39	193.67	**467.04**
	10	163.25	139.16	335.88	234.94	309.84	122.91	221.73	268.28	101.77	**345.95**

5.3.2 Performance of multi-objective key player identification algorithm on the Immunization problem

In a population in which a disease is spreading, finding the best subset of nodes (constrained by a budget) to immunize is a problem of immense interest. This problem can be modeled using a network, where the individuals in the population are denoted by nodes and the relationships among the individuals are modeled by the edges. It has been accepted that the strategies that use key players to immunize are highly efficient for this problem [25]. Degree centrality and Betweenness centrality based immunization strategies and their variants are shown to be very efficient for scale-free network models and some real networks [84].

In this section, we evaluate the performance of the multi-objective key player identification technique for this problem.

Multiple computational models have been proposed over the years to explain the process of disease spread [89]. For example, in SIR model, a population of N individuals is divided into three states: susceptible (S), infected (I), and recovered (R). Initially, all the nodes are in the state S. The disease starts to spread starting from a single infected node. At each time step, the disease spreads to infected nodes' neighbors with a probability β. Each infected node has a probability of recovering γ at each time step. The model stabilizes when the number of infected nodes in the network reaches 0. In the immunization problem, the goal is to identify the best subset of nodes to be immunized, to control the disease spread. The immunization process starts after a certain delay from the start of disease spread. We select nodes to immunize based on different key player identification algorithms, and compare the effect of each immunization strategy on disease spread. Two measures are considered in deciding the effectiveness of immunization strategies:

i. The average time to stabilize; and

ii. The average infection probability of all the nodes in the network.

To evaluate these averages we repeat the entire experiment 1000 times and use $\beta = 0.3$ and $\gamma = 0.05$.

Leave-k-out approach for multi-objective optimization

First, similar to the Section 4.5.1, in Table 5.8 we compare the approaches that have been proposed to select solutions from the Pareto set (in Section 3.3.1) against the novel approach we introduced (*Leave-k-out* approach).

Similar to the results in Section 4.5.1, no method significantly outperforms other methods. Hence, we conclude that, all the considered methods for selecting best solutions from

Table 5.8: The comparison of Pareto set pruning approaches on the Immunization problem

Method	Average time to stabilize		Average infection probability		Running time (s)	
	ca-GrQc	PGP	ca-GrQc	PGP	ca-GrQc	PGP
Leave-k-out (k=1)	137.40	259.43	0.26	0.34	195.17	386.24
Leave-k-out (k=2)	138.65	262.34	0.29	0.36	143.14	295.32
Leave-k-out (k=3)	140.11	276.51	0.32	0.40	107.32	211.41
Weighted sum	137.93	260.32	0.28	0.34	311.58	538.75
Average Ranking	137.99	261.97	0.28	0.35	314.28	540.11
Maximum Rank	139.56	265.21	0.30	0.36	317.24	540.41
Cluster centers	140.04	263.84	0.27	0.36	387.56	574.32
Ideal points	139.43	262.71	0.27	0.35	390.11	577.18
Angle based pruning	138.38	259.84	0.26	0.34	401.46	603.21
Favor Relation	138.35	260.62	0.27	0.35	832.24	984.53
K-optimality	137.27	259.21	0.26	0.33	427.42	611.83

the Pareto set perform equally well, when evaluated based on the performance on the *Immunization* problem,

But the *Leave-k-out* approach outperforms all other approaches significantly when the running time to identify the solutions (from the Pareto space) is considered $(p < 0.001)$. The reduced running time of the *Leave-k-out* approach occurs due to the reduction of objectives in the multi-objective optimization. In the above experiment, *Leave-k-out (k=1)*, *Leave-k-out (k=2)*, and *Leave-k-out (k=3)* requires 4, 3 and 2 objective optimization in the first step respectively. All the other approaches require 5 objective optimization.

To evaluate the quality of the solutions obtained by the *Leave-k-out* approach, we consider the probability that the solutions identified by the *Leave-k-out* approach are also the solutions in the Pareto front of the unaltered multi-objective optimization. Table 5.9 shows the results. All the results shown are averages over 30 trials.

According to the results, the solutions obtained by *Leave-k-out (k=1)* are always found in the Pareto front for the 5 objective optimization of the *Immunization* problem, for all four networks considered. The solutions obtained by *Leave-k-out (k=2)*, and *Leave-k-out (k=3)* have probabilities 0.91 and 0.84 respectively of being in the Pareto front of the

Table 5.9: Probability of the solutions identified by the *Leave-k-out* approach also being solutions in the Pareto front of the original multi-objective optimization (Immunization problem)

	ca-GrQc	PGP	ca-HepPh	Gplus
Leave-k-out (k=1)	1.0	1.0	1.0	1.0
Leave-k-out (k=2)	0.91	0.92	0.90	0.91
Leave-k-out (k=3)	0.84	0.86	0.81	0.83

5 objective optimization for the ca-GrQc network. This indicates that the quality of the solutions decreases when more objectives are left out from the initial optimization. This is consistent with the results obtained for the *EIL* problem in Section 4.5.1. A similar pattern is observed for the PGP, ca-HepPh and Gplus networks as well.

When *Leave-k-out (k=1)* and *Leave-k-out (k=3)* are compared, for a 2% decrease in performance (average time to stabilize) a 45% reduction in computational time is achieved when *Leave-k-out (k=3)* is used for the ca-GrQc network. Hence *Leave-k-out (k=3)* is recommended for this application.

Performance comparison of key node identification approaches on the Immunization problem

Table 5.10 shows the results for average time to stabilize for the ca-GrQc network, and Table 5.11 shows the results for average infection probability for the ca-GrQc network. Similar to the *EIL* problem we assume that the set of qualities key players should have are to be measured by Degree Centrality, Betweenness Centrality, Closeness Centrality, Eigenvector Centrality and PageRank, and use *Leave-k-out* approach with ($k = 2$) to select the best solutions from the Pareto set. The performance of the key player sets identified on the original network is better than the performance of the key player sets identified on the sampled network. Hence the performance of key player sets obtained on the sampled network by single objective key player identification methods is not shown in Tables 5.10 and 5.11.

The best value obtained for each case is highlighted in the tables 5.10 and 5.11. Accord-

Table 5.10: Average Time to stabilize

Delay	Fraction Immunized	DC	EC	BC	CC	PR	PCC	ITP	ITN	KS	MO
2	0.05	162.42	166.71	156.58	168.32	154.79	165.27	171.56	147.90	171.74	**137.67**
	0.10	154.27	160.54	131.84	148.14	132.92	154.18	166.72	133.38	165.08	**119.24**
	0.15	145.98	154.28	82.31	141.44	82.71	155.50	133.76	121.84	157.80	**76.07**
	0.20	87.46	148.04	82.83	124.54	**69.07**	155.13	82.89	119.78	158.70	69.76
	0.25	70.33	133.75	71.14	101.86	65.60	136.05	77.69	108.38	133.56	**51.08**
3	0.05	166.01	164.90	155.56	161.71	161.56	163.59	173.47	148.60	167.21	**138.71**
	0.10	159.04	163.56	140.07	159.70	141.35	175.39	160.15	143.26	165.06	**127.40**
	0.15	152.54	163.09	108.67	154.41	104.36	160.75	158.46	135.75	175.63	**92.75**
	0.20	107.77	160.39	96.33	140.67	**88.80**	157.19	104.95	127.23	148.27	89.09
	0.25	92.90	151.26	90.59	121.68	83.23	154.27	85.47	130.63	141.86	**75.12**
4	0.05	166.09	165.85	164.61	164.61	163.30	168.87	168.40	155.67	168.66	**141.41**
	0.10	164.51	166.12	149.98	163.13	155.90	166.33	161.99	153.33	166.01	**137.98**
	0.15	156.38	159.83	122.42	155.17	126.14	160.25	149.25	142.12	159.76	**114.82**
	0.20	130.41	157.07	115.28	142.71	115.82	159.25	124.14	136.28	154.34	**92.67**
	0.25	113.91	152.06	106.58	136.82	109.28	156.09	109.89	139.88	155.58	**86.21**
5	0.05	167.28	167.44	161.46	161.36	165.38	168.39	170.58	170.57	167.20	**147.28**
	0.10	164.02	167.89	152.77	160.20	155.43	159.56	173.98	155.82	172.23	**131.87**
	0.15	157.62	167.69	134.33	160.73	136.86	159.94	153.89	143.74	163.04	**122.98**
	0.20	138.82	159.07	132.90	149.67	129.11	160.50	141.88	142.09	161.72	**118.65**
	0.25	133.33	152.65	128.48	148.40	124.38	161.79	138.69	147.80	150.18	**100.62**

ing to the results, both for average infection probability and average time to stabilize, the multi-objective key players identified on 50% sample of the original network achieve the significantly better results ($p < 0.05$) and the best result among all key player identification algorithms for 90% of the cases.

More sophisticated approaches have been proposed to address the problem of maximizing information diffusion in social networks [64, 16] and the problem of immunization [104, 101]. But these approaches require more information about the network (such as node, edge attributes) and are proposed to solve these specific problems. Our goal in this study was to propose a new strategy to identify key players that is general, and can be applied to multiple applications of interest.

Table 5.11: Average infection probability

Delay	Fraction Immunized	DC	EC	BC	CC	PR	PCC	ITP	ITN	KS	MO
2	0.05	0.75	0.80	0.57	0.73	0.60	0.80	0.80	0.57	0.85	**0.42**
	0.10	0.57	0.71	0.24	0.51	0.25	0.70	0.59	0.42	0.78	**0.17**
	0.15	0.30	0.61	0.05	0.36	0.05	0.67	0.24	0.29	0.63	**0.04**
	0.20	0.05	0.52	0.04	0.18	0.05	0.56	0.08	0.23	0.49	**0.03**
	0.25	0.03	0.39	0.03	0.08	0.03	0.47	0.03	0.10	0.29	**0.02**
3	0.05	0.78	0.81	0.62	0.73	0.65	0.81	0.80	0.59	0.84	**0.47**
	0.10	0.63	0.74	0.32	0.59	0.32	0.79	0.60	0.48	0.78	**0.26**
	0.15	0.37	0.67	0.14	0.46	0.12	0.68	0.40	0.34	0.74	**0.09**
	0.20	0.14	0.60	0.11	0.30	**0.081**	0.60	0.13	0.31	0.51	0.086
	0.25	0.09	0.49	0.08	0.19	0.08	0.53	0.08	0.18	0.35	**0.06**
4	0.05	0.81	0.82	0.69	0.77	0.69	0.83	0.80	0.62	0.83	**0.51**
	0.10	0.70	0.79	**0.42**	0.66	0.46	0.76	0.67	0.54	0.80	0.45
	0.15	0.47	0.71	0.27	0.55	0.26	0.69	0.45	0.41	0.70	**0.17**
	0.20	0.29	0.61	0.21	0.39	0.21	0.65	0.27	0.37	0.58	**0.11**
	0.25	0.21	0.56	0.18	0.33	0.19	0.58	0.22	0.30	0.49	**0.07**
5	0.05	0.83	0.84	0.72	0.78	0.74	0.84	0.82	0.71	0.86	**0.59**
	0.10	0.73	0.83	0.54	0.70	0.53	0.76	0.78	0.60	0.86	**0.46**
	0.15	0.56	0.77	0.40	0.65	0.411	0.71	0.56	0.47	0.76	**0.38**
	0.20	0.44	0.70	0.36	0.54	0.34	0.68	0.36	0.46	0.69	**0.25**
	0.25	0.38	0.64	0.34	0.49	0.33	0.64	0.37	0.39	0.55	**0.18**

5.4 Concluding Remarks

As the size and the complexity of the networks increase, so do the running times of the network analysis measures. The proposed multi-objective approach to key player identification depends on the computational complexity of individual network centrality measures and on the computational complexity of evolutionary optimization algorithm employed. Hence we investigated how network sampling can be used to reduce the running times of key player identification without compromising much on the quality of key nodes obtained. We proposed the idea of degree centrality based sampling to reduce the running time of the key node identification problem. First, we showed that the degree centrality based sampling method manages to retain most of the original network's key nodes in the sample compared to other sampling methods. Then we used the multi-objective key player sets obtained on degree centrality based sampled networks to address two well known problems, viz., *Eventual Information Limitation (EIL)* problem and *Immunization* problem. The results show that the multi-objective key player sets identified on sampled networks perform better than single objective key player sets identified by applying the algorithms on the entire network.

CHAPTER 6 IMPROVING NETWORK ROBUSTNESS USING KEY EDGES

Networked infrastructure systems such as the road network, airline network, power grid, etc. play an important role in our day to day activities. Hence maintaining the functionality of these systems during natural damages or attacks on their components is a critical concern. Quantifying the resilience of the network can be done in multiple ways, depending on the application and the network properties of interest; hence there exists no unique definition for network robustness. Thus, multiple network robustness measures have been introduced to evaluate the capability of a system to withstand such failures or attacks. All such measures aim to capture features such as: (1) Connectivity - robust networks are expected to remain connected even when a set of nodes or edges fail during targeted or natural node/edge failures, (2) Distance - distances between the nodes of robust networks should remain minimally affected during node/edge failures, and (3) Network properties - the network properties such as degree distribution and distance distribution should change very little for robust networks.

Improving the robustness of infrastructure systems is an important research problem. Adding extra edges constraining to a budget and degree-preserving edge rewiring are two techniques that have been proposed to address this problem [116, 20, 21, 109, 55, 95]. When edges are added to a network, the properties of the network change. The amount of change depends on the importance of the set of edges added to the network. In this study, we assume that upon addition a set of key edges should maximally improve the network robustness.

In this chapter we address the problem of ‘Given a network and a budget, how should a set of key edges be selected to be added to the network in order to maximally improve multiple robustness measures of interest’. In Section 6.1 we discuss the network robustness measures that have been proposed and widely used. Then in Section 6.2 we analyze the properties of these robustness measures and identify the similarities and dissimilarities using correlation analysis. In Section 6.3, we introduce our algorithm to optimize multiple robustness measures of interest to improve the overall robustness of a network. In Section 6.4, we provide the experimental results which show the improvements in multiple robustness measures when the new edges are added using our algorithm.

6.1 Robustness measures for networks

We consider three categories of network robustness measures that have been proposed and are widely used in the literature.

6.1.1 Measures based on the eigenvalues of the adjacency matrix

Let A be the adjacency matrix of the network $G = (V, E)$ with n nodes, and let $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq ... \geq \lambda_n$ be the set of eigenvalues of A.

1. *Spectral radius (SR)* : The largest or the principal eigenvalue λ_1 is called the spectral radius. This has been used as a measure of quantifying network robustness in multiple studies [20, 71, 112, 113]. SR is inversely proportional to the epidemic threshold of a network [19].

2. *Spectral gap (SG)* : The difference between the largest and the second largest eigenvalues $(\lambda_1 - \lambda_2)$ is called the spectral gap. This has also been used to measure the robustness of the graph against attacks [20, 80, 122]. Spectral gap is related to the expansion properties of the graph; networks with good expansion properties provide excellent communication platforms due to the absence of bottlenecks [117].

3. *Natural connectivity (NC)* : Denoted by $\bar{\lambda}$, natural connectivity is defined as follows:

$$\bar{\lambda} = \ln\left(\frac{1}{n}\sum_{j=1}^{n} e^{\lambda_j}\right)$$

   and is widely used as a measure of robustness in complex networks [20, 63, 21]. Natural connectivity characterizes the redundancy of alternative paths in the network by quantifying the weighted number of closed walks of all lengths [21]. Clearly this is an important measure, because redundancy of routes between the nodes ensures that communication between nodes remain possible during an attack to the network. A network created by optimizing the natural connectivity is found to exhibit a roughly 'eggplant-like' topology, where there is a cluster of high-degree nodes at the head and other low-degree nodes scattered across the body of the 'eggplant'[95].

6.1.2 Measures based on the eigenvalues of the Laplacian matrix

The topology of a network G with n nodes can also be represented by the $n \times n$ Laplacian matrix $\mathcal{L} = D - A$, where $D = diag(d_u)$ and d_u is the degree of node u. Let the set of eigenvalues of $\mathcal{L}$ be $\mu_1 = 0 \leq \mu_2 \leq \mu_3 \leq ... \leq \mu_n$; these are used to define the following measures:

1. *Algebraic connectivity (AC)* : This is the second smallest eigenvalue of the Laplacian matrix (μ_2). The algebraic connectivity is 0 if the network is disconnected and $0 < \mu_2 \leq n$ when the network is connected [42]. The larger the AC, the more difficult

it is to cut a graph into disconnected components [2], hence this has been used by many studies to determine the robustness of networks [62, 109, 20].

2. *Normalized Effective resistance (nER)*: Introduced in [37], nER is defined as,

$$nER = \frac{n-1}{\left(n \sum_{i=2}^{n} \frac{1}{\mu_i}\right)}$$

The usefulness of this measure can be seen when the network is viewed as an electrical circuit with edges representing a resistor with electrical conductance equal to the edge weight. The effective resistance (R_{vu}) between a pair of nodes u and v is small when there are many paths between nodes u and v with high conductance edges, and R_{uv} is large when there are few paths, with lower conductance, between nodes u and v [47]. Effective resistance is equal to the sum of the (inverse) non-zero Laplacian eigenvalues and has been used in multiple studies to define network robustness [20, 37].

'Network criticality' is a similar robustness metric defined to capture the effect of environmental changes such as traffic variation and topology changes in networks [8, 111].

6.1.3 Measures based on other properties

1. *Harmonic diameter (HD)*: This is defined as follows:

$$HD(G) = \frac{n(n-1)}{\left(\sum_{u \neq v \in V} \frac{1}{d(u,v)}\right)}$$

where n is the number of nodes in the network and $d(u,v)$ is the shortest distance between the nodes u and v [82]. HD has been used to evaluate network robustness in multiple studies [11, 12]. This measure is analogous to the average distance between all the nodes, but better because this can also be applied to disconnected networks. For ease of comparison with other measures, we use the reciprocal of the Harmonic diameter (rHD) in this study, which increases with robustness.

2. *Size of the largest connected component (LCC)*: This measure identifies the size of the largest component during all possible malicious attacks. LCC is defined as follows:
$$R = \frac{1}{n+1}\sum_{Q=0}^{n} s(Q)$$
where n is the number of nodes in the network and $s(Q)$ is the fraction of nodes in the largest connected cluster after attacking Q nodes. LCC was proposed in [55] and is widely used [102, 110, 121, 124] as a robustness measure in networks. The normalization factor $\frac{1}{n+1}$ ensures that the robustness of networks with different sizes can be compared. The attacks often consist of a certain fraction of node attacks, and after the attack, the measure identifies the number of nodes in the largest connected component. It has been found that the robust networks that optimize this measure form a unique 'onion-like' structure consisting of a core of highly connected nodes hierarchically surrounded by rings of nodes with decreasing degree [55].

3. *Clustering coefficient (CC)*: The abundance of triangles in the network is identified by the clustering coefficient [118]. The clustering coefficient of a network is calculated based on the local clustering coefficient of each node. The clustering coefficient of the node u is defined as,

$$CC_u = \frac{number\ of\ triangles\ connected\ to\ node\ u}{number\ of\ triples\ centered\ around\ node\ u}$$

where a triple centered around node u is a set of two edges connected to node u. The overall clustering coefficient of the network is calculated as the average CC_u. A high clustering coefficient indicates high robustness, because the number of alternative paths grows with the number of triangles [38].

Other robustness measures have also been proposed in literature, e.g, vertex/edge connectivity, network diameter, average distance between the nodes, vertex/edge betweenness and number of spanning trees. These measures are excluded in this study due to poor performance in some trivial cases or high computational cost needed for real world large networks [38].

In the following section we discuss some of the properties of the aforementioned network robustness measures.

6.2 Properties of network robustness measures

In this section we compare the aforementioned robustness measures using three approaches:

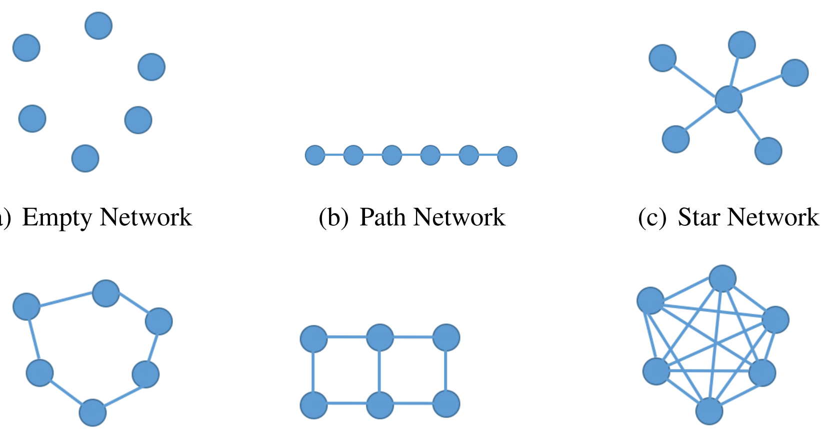
i. Robustness values of a few small networks are calculated and compared.

ii. The change in these robustness measures upon addition of new edges to the network are calculated and compared.

iii. The similarities and dissimilarities of the robustness measures are compared using the correlation of these measures for a set of generated networks that follow the power law degree distribution.

6.2.1 Analysis of trivial networks

The trivial networks that we considered are shown in Figure 6.1. The networks are ordered by increasing robustness intuitively, i.e., the network 6.1(a) is the least robust, the network


(a) Empty Network (b) Path Network (c) Star Network

(d) Ring Network (e) Grid Network (f) Fully connected Network

Fig. 6.1: Six trivial networks considered for robustness calculation; the networks are arranged in the increasing order of robustness assesed intutively.

6.1(b) is more robust than network 6.1(a), the network 6.1(c) is more robust than 6.1(b), etc. , and the network 6.1(f) is the most robust.

Table 6.1 shows the robustness values obtained by each robustness measure discussed in Section 6.1 for each trivial network that we consider. The summary of the results is as follows:

i. NC and nER orders the networks in the expected order.

ii. rHD and AC also order the networks correctly, but fail to distinguish between some of the trivial networks.

iii. CC gives a value of 0 to all networks with no triangles, and evaluates the empty network to be as robust as the grid network.

iv. LCC, SR and SG order the networks differently than our intuition.

v. LCC gives the same robustness value to both the ring and grid networks, defying intuition. In addition, the star network gets a low LCC value than the path network.

vi. SR and SG identify the networks which enable fast communication as robust networks, thus star network gets a high robustness value.

vii. All the robustness measures considered identify the empty network as the least robust network and the fully connected network as the most robust.

Table 6.1: Robustness values of the trivial networks

Network	rHD	LCC	CC	SR	SG	NC	AC	nER
Empty	0	0	0	0	0	0	0	0
Path	0.58	0.33	0	1.80	0.55	0.71	0.27	0.14
Star	0.66	0.17	0	2.24	2.24	0.81	1.0	0.20
Ring	0.66	0.37	0	2.0	1.0	0.83	1.0	0.28
Grid	0.70	0.37	0	2.41	1.41	1.01	1.0	0.35
Full	1.00	0.50	1.0	5.0	6.0	3.22	6.0	1.0

6.2.2 Behavior of the Robustness measures for a single edge addition to the network

Multiple studies have used edge addition to improve the network robustness [21, 116]. Intuitively when an edge is added to the network, the overall robustness of the network should improve. In this section, we examine whether this hypothesis is true for all the robustness measures when a single edge is added to the network.

Among all the aforementioned robustness measures, rHD, NC, nER and SR monotonically increase upon an addition of an edge (u, v) regardless of the location of the new edge. A brief argument is presented below to support the above claim.

- rHD - When a new edge (u, v) is added to a network, the shortest distance between the two nodes u and v is reduced. Thus rHD will necessarily increase.
- NC - Natural connectivity is proportional to the weighted sum of numbers of closed walks in the network. When a new edge is added to a connected network, it will create a new cycle in the network, thus the number of closed walks in the network will increase. Thus NC will monotonically increase upon edge addition.

- nER - Suppose nER between the nodes u and v before adding the new edge was R_{uv}. After the new edge is added, the nER between u and v becomes $\left(1+\frac{1}{R_{uv}}\right)^{-1}$. Since, $\left(1+\frac{1}{R_{uv}}\right)^{-1} < R_{uv}$, nER increases with an edge addition.

- SR - Consider $G' = G - (u, v)$ and x be the eigenvector of G' corresponding to the leading eigenvalue $\lambda_1(G')$. Then, the following inequality holds, if $x_u x_v > 0$:

$$\lambda_1(G') = x^t A(G')x < x^t A(G)x = \lambda_1(G),$$

  since $x^t A(G')x = x^t A(G)x - 2x_u x_v$.

The monotonic increase of robustness with edge addition is not a property of other robustness measures that were mentioned in Section 6.1. CC increases only when the newly added edge creates a triangle in the network, and the difference between the CC in ring and grid networks (shown in 6.1) is an example for this. AC also does not necessarily increase with an edge addition. The difference between the AC values between the ring network and the grid network in Table 6.1 provides an example for this as well. The difference in SG between a star network and a network created by adding an edge between two peripheral nodes gives an example for a case where the SG decreases when an edge is added. LCC depends on the order in which the attacks occur on the network. Thus an edge addition does not always guarantee increase in every robustness measure.

6.2.3 Correlation of robustness measures

In this section, we study similarities in the overall behavior of the robustness measures that were discussed in Section 6.1, using Pearson's correlation coefficient. A high correlation between two measures suggests that a network that shows high robustness in terms of one measure would also show high robustness in terms of the other measure as well.

To evaluate the correlations among the robustness measures, 100 scale free networks were generated with number of nodes in range $[500, 5000]$ and with power law parameters

Table 6.2: Correlation of the robustness measures

	rHD	**LCC**	**CC**	**SR**	**SG**	**NC**	**AC**	**nER**
rHD	1							
LCC	-0.41	1						
CC	0.90	-0.61	1					
SR	0.88	-0.65	0.88	1				
SG	0.89	-0.58	0.98	0.87	1			
NC	0.90	-0.62	0.99	0.89	0.98	1		
AC	0.86	-0.34	0.75	0.86	0.78	0.76	1	
nER	0.37	0.32	0.02	0.28	0.09	0.04	0.69	1

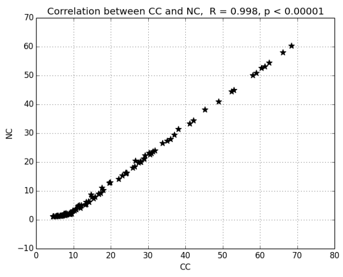
in range $[2.0, 3.0]$. For each of the generated networks, the robustness values were calculated. Then, the Pearson product-moment correlation coefficient was calculated between the robustness measures. The correlation coefficients are shown in Table 6.2. [1]

According to the results, some of the robustness measures are highly correlated. Some of these highly correlated pairs include (CC, NC), (SG, NC) and (SG, CC) ($p < 0.001$). Also, the three robustness measures that are calculated using the eigenvalues of the adjacency matrix (spectral radius, spectral gap and natural connectivity) are highly correlated.

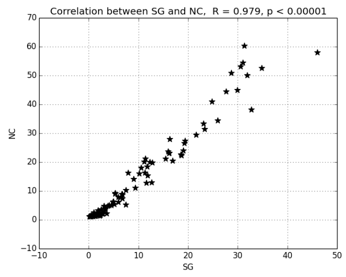
Some robustness measures are highly uncorrelated. For example, the pairs (CC, nER), (NC, nER) and (SG, nER) show this behavior ($p > 0.5$). The correlation of some of the robustness measure pairs are shown in Figures 6.2(d)-6.2(f).

Interestingly, LCC negatively correlates with most other robustness measures (except for nER). The negative correlation suggests that when the robustness of the network is increased in terms of LCC, the robustness in terms of other measures will not increase. The observed negative correlation can be explained as follows. Consider improving the LCC measure. As LCC is focused on keeping most of the nodes in a single connected component during node attacks made in the order of degree centrality, new edges (that get added when optimizing LCC) connect nodes with low degree and in different communities in the network. However such edge addition decreases, for example CC (and other measures),

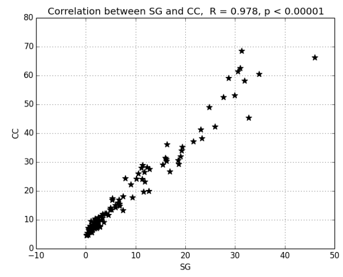
[1]Similar results were obtained when the experiments were carried out for 100 generated scale free networks by, (1). fixing the number of nodes and changing power law parameter in the aforementioned range and, (2). changing the number of nodes in the aforementioned range and fixing the power law parameter.



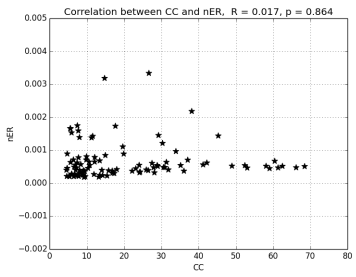
(a) CC vs NC



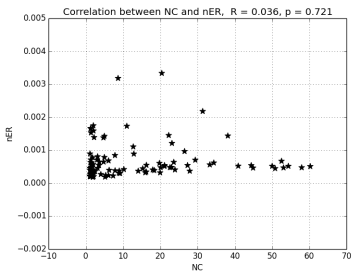
(b) SG vs NC



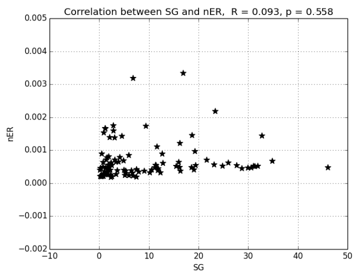
(c) SG vs CC



(d) CC vs nER



(e) NC vs nER



(f) SG vs nER

Fig. 6.2: Correlation plots between the robustness measures

because although the number of edges increase, the number of triangles in the network mostly remain unchanged. The number of triangles are less likely to increase, because: (1) the number of edges connecting different communities is small, and (2) the nodes to which the edges are added have low degrees.

6.3 Multi-objective definition of robustness

Multiple studies have focused on improving the robustness of a network by optimizing a single robustness measure [116, 117, 47, 109, 21, 112, 20]. But, the low correlation among some of the robustness measures (shown in Section 6.2.3) suggests that when a single measure is optimized, it does not guarantee the improvement of robustness in terms of other measures. We argue that edges should be added to a network in a manner such that not one, but multiple robustness measures improve. In this section, we propose a methodology to improve multiple uncorrelated robustness measures by adding new edges to the network.

We select three uncorrelated robustness measures (largest connected component (LCC), spectral gap (SG) and normalized effective resistance (nER)), one each from the three categories of robustness measures discussed in Section 6.1. Our goal is to improve all three of these measures in the network by edge addition. We formulate the problem as a multi-objective maximization problem : *Find the set of k edges that can increase all three robustness measures the most.*

We represent the network as a bit string, in which each possible edge that can be added to the network is assigned an index. The number of bits in the bit string is equal to the number of possible edges (m) that can be added to the network. Initially, before any extra edge is added to the network, the bit string consists of all 0s. When a certain edge is selected to be added to the network, the bit value corresponding to the index of the selected edge will be changed to 1.

The key steps of the NSGA-II algorithm to identify the k edges to add to the network are as follows:

i. Initial population - In each individual (bit string) in the initial population, k random bits are assigned the value of 1 to represent that they are selected to be added to the network, and the remaining $(m - k)$ bits are assigned 0, where m is the total number of edges that can be added to the network.

ii. Fitness - To calculate the fitness value of each individual, we first add the selected set of k edges to the initial network. Then, the three robustness measures (LCC, SG and nER) are calculated for the amended network.

iii. Crossover - One point crossover is applied to a fraction P_c of selected individuals to generate offspring.

iv. Mutation - Mutation is performed with probability P_m by inverting two bits of different values.

The number of solutions obtained depends on the network on which the optimization is performed and the objectives selected.

6.3.1 Fast calculation of robustness measures

The computation of the three robustness measures that we consider is costly for large networks. Thus we use approximation techniques and the results of Matrix Perturbation Theory for fast calculation of robustness measures.

1. *Size of the largest connected component (LCC):*

   Computing LCC requires calculation of the fraction of nodes in the largest connected cluster after attacking nodes in the order of degree centrality. In many real networks, the degree distribution follows the power law. Hence, the attacks made on the high degree nodes have the biggest impact on the network. We approximate LCC by attacking only the top $l\%(<< n)$ nodes with the highest degree centrality. For 100 generated scale-free networks with number of nodes in range $[500, 5000]$ and scale-free parameter in range $[2.0, 3.0]$, the LCC calculated by removing all n nodes in the network has a correlation of $0.87(p < 0.0001)$ with the LCC approximated by removing only the top 20% degree centrality nodes. This approximation reduces the running time of LCC by 79.9% on average. Hence, in our experiments we approximate the LCC by attacking the top 20% nodes in the network in the order of degree centrality.

2. *Spectral gap (SG)*

   For a perturbation ΔA in the adjacency matrix A of the original network G, the new eigenvalues and eigenvectors of the new network G' can be approximated [107, 20]. The update to the i^{th} eigenvalue can be written as $\Delta\lambda_i \approx \boldsymbol{x_i}^T \Delta A \boldsymbol{x_i}$ and the approximated change in the i^{th} eigenvector is $\Delta \boldsymbol{x_i} \approx \sum_{j=1, j\neq i}^{n} \left(\frac{\boldsymbol{x'_j} \Delta A \boldsymbol{x_i}}{\lambda_i - \lambda_j} \boldsymbol{x_j} \right)$. Thus the change in SG when a new edge (u, v) is added can be approximated by the expression,

$$\boldsymbol{x_1^T} \Delta A \boldsymbol{x_1} - \boldsymbol{x_2^T} \Delta A \boldsymbol{x_2} = 2(\boldsymbol{x_{1u}} \boldsymbol{x_{1v}} - \boldsymbol{x_{2u}} \boldsymbol{x_{2v}}),$$

where $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are the eigenvectors corresponding to two largest eigenvalues λ_1 and λ_2, and $\boldsymbol{x_{ij}}$ denotes the j^{th} element of the i^{th} eigenvector. Since we avoid calculating all the eigenvalues of a large adjacency matrix, the running time is substantially reduced.

3. *Normalized Effective Resistance (nER)*

   Effective resistance is equal to the sum of reciprocals of the non-zero Laplacian eigenvalues, and can be approximated by the first $l - 1$ non-zero eigenvalues instead of all $n - 1$ of them [37]. According to matrix perturbation theory, when an edge (u, v) is added to a network, the change in its Laplacian eigenvalue μ_t can be written as, $\Delta\mu_t = \boldsymbol{v_t^T} \Delta\mathcal{L} \boldsymbol{v_t} = (\boldsymbol{v_{tu}} - \boldsymbol{v_{tv}})^2$, where μ_t is the t^{th} eigenvalue of the Laplacian matrix $\mathcal{L}$, $\boldsymbol{v_t}$ is the corresponding eigenvector of μ_t and $\boldsymbol{v_{ti}}$ corresponds to the i^{th} element of the t^{th} eigenvector.

   Using this eigenvalue approximation and matrix perturbation theory, the change in nER when an edge (u, v) is added can be written as,

$$\Delta nER \approx \frac{l-1}{n(\frac{1}{\mu_2+\Delta\mu_2} + \frac{1}{\mu_3+\Delta\mu_3} + ... + \frac{1}{\mu_l+\Delta\mu_l})} - nER$$

$$\Delta nER \approx \frac{l-1}{n} \left(\sum_{i=2}^{l} \frac{1}{\mu_i + (\boldsymbol{v_{iu}} - \boldsymbol{v_{iv}})} \right)^{-1}$$

6.3.2 Selecting solutions from multi-objective optimization

The Pareto set usually contains a large number of solutions. For example, in identifying 10 new edges to add to the EuroRoad network to maximize all 3 objectives, the Pareto surface

Table 6.3: Statistics and description of the networks used

Network	**Nodes**	**Edges**	**Description**
Euroroad[2]	1174	1417	International road network in Europe. The nodes represent cities and an edge indicate the cities are connected by a road.
US airports[3]	1574	28236	The network of flights between the US airports in 2010. Each edge represents a connection from one airport to another.
OpenFlights[4]	2939	30501	The network of flights between airports in the world. An edge represents a flight from one airport to another.
US power grid[5]	4941	6594	The power grid of the Western States of the US. A node is either a generator or a power station, and an edge represents a power line.

contained 91 solutions i.e., 91 sets of 10 edges. Since there are too many solutions for decision making, we use the *Leave-one-out* approach discussed in Section 4.4.1.

6.4 Experimental Results

We use four commonly used real world network datasets in our experiments. A brief description of the datasets is provided in Table 6.3.

6.4.1 Improving robustness by edge addition

For each case of robustness optimization, we provide four multi-objective optimization solutions. The first value corresponds to the average of all solutions in the Pareto surface.

[2]Source: http://konect.uni-koblenz.de/networks/subelj_euroroad
[3]Source: http://konect.uni-koblenz.de/networks/opsahl-usairport
[4]Source: http://konect.uni-koblenz.de/networks/opsahl-openflightsd
[5]Source: http://konect.uni-koblenz.de/networks/opsahl-powergrid

The other three values correspond to the solutions obtained by the *Leave-k-out* approach (k=1) (Leave-one-out approach).

1. $\mathcal{O}_{ave}$ - Represents the average value obtained by all the solutions in the Pareto front of the 3 objective optimization.

2. $(\mathcal{O}-1)_{nER}$ - In this case we first perform the optimization on SG and LCC. Then from the non-dominated solutions obtained, the solution that maximizes nER is selected.

3. $(\mathcal{O}-1)_{LCC}$ - The initial optimization is performed on SG and nER. Then from the non-dominated solutions obtained, the solution that maximizes LCC is selected.

4. $(\mathcal{O}-1)_{SG}$ - First, the optimization on LCC and nER is performed. Then from the non-dominated solutions obtained, the solution that maximizes SG is selected.

We use the solutions obtained by multiple other edge addition methods to compare the results. First we consider single objective optimization to improve network robustness. We obtain sets of edges that optimize SG, LCC and nER respectively.

Then we consider the following heuristic approaches to add edges to the network, which would also improve network robustness.

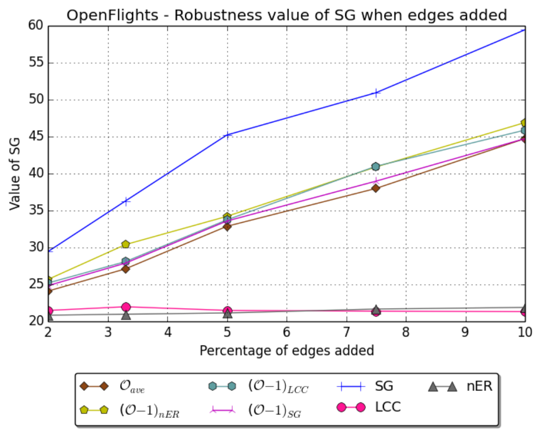
i. $Rich-Rich$: The edges are added among the nodes with high degree.

ii. $Poor-Poor$: The edges are added among the nodes with lowest degree.

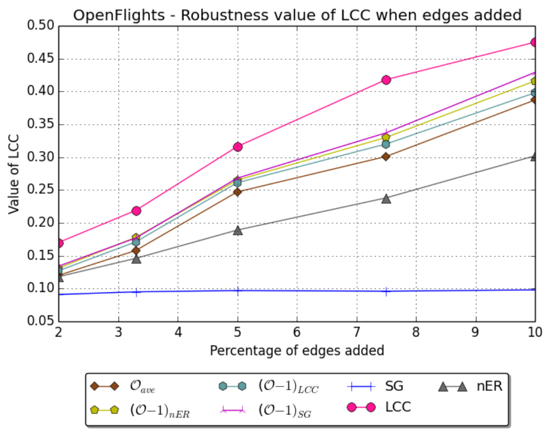
iii. $Rich-Poor$: The edges are added between the nodes with high degree and nodes with low degree.

iv. $Random$: In this case we add edges randomly to the network.

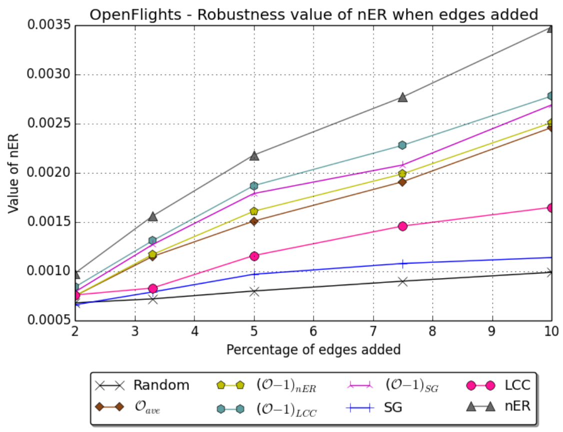
We present the results in Figures 6.3 and 6.4. In Figure 6.3, the robustness improvement obtained by the proposed multi-objective approach is compared with the robustness



(a) Value of SG after edges added



(b) Value of LCC after edges added



(c) Value of nER after edges added

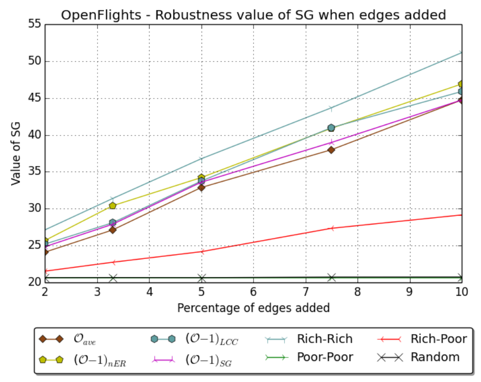
Fig. 6.3: Robustness improvement in OpenFlights network - Comparison between multi-objective approach and single objective approaches

improvement obtained by optimizing single robustness measures. In Figure 6.4, we compare the robustness improvement by the multi-objective approach with the heuristic edge addition methods. In the figures, the Y axis corresponds to the robustness measure achieved by the network, upon edge addition. Higher values along the Y axis corresponds to more robust networks. The X axis represents the percentage of new edges added to the network.

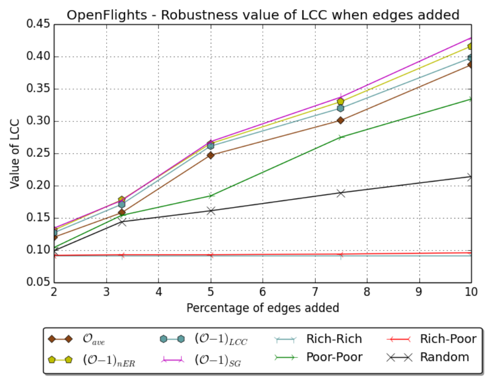
In Figure 6.3(a), we show how the robustness value of SG has changed with the use of different edge addition algorithms. As expected, the solution obtained by optimizing SG gives the best improvement in SG compared to the other algorithms. What is to be noted here is the poor improvement of SG given by the solutions that were optimized for LCC and nER. A similar pattern is seen in Figure 6.3(b) and Figure 6.3(c) as well. In Figure 6.3(b) where we plot the value of LCC with edge addition, the best performance is shown by the solution that was obtained by optimizing LCC, whereas the solutions that were obtained by optimizing SG and nER perform poorly. In Figure 6.3(c) where we plot the value of nER with edge addition, the best performance is shown by the solution that was obtained by optimizing nER and the solutions that were obtained by optimizing SG and LCC do not perform well.

The solutions obtained by multi-objective optimization do not perform the best in every case, but perform 'well' in all the cases. For example, the solution obtained by $(\mathcal{O}-1)_{nER}$ performs 3^{rd} best in optimizing SG and LCC and performs 4^{th} best in optimizing ER among all the methods that we considered. But the network created by adding a set of new edges to optimize a single objective robustness measure (such as SG), will perform well only when the optimized robustness measure is concerned. For example, the network created by optimizing SG performs the best when the value of SG is considered (Figure 6.3(a)), but performs really poorly when the other two robustness measures are considered (Figures 6.3(b) and 6.3(c)).

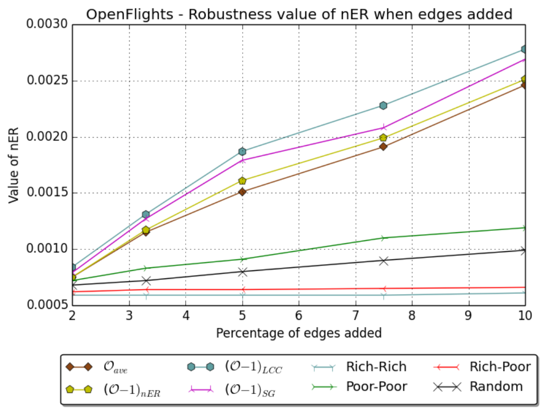
The solutions obtained by *Leave-k-out* approach perform slightly better than the average of solutions of $\mathcal{O}$ objective optimization in each case. This is because in $\mathcal{O}$ objective



(a) Value of SG after edges added



(b) Value of LCC after edges added



(c) Value of nER after edges added

Fig. 6.4: Robustness improvement in OpenFlights network - Comparison between multi-objective approach and heuristic approaches

optimization, when an extra objective is added to the optimization, we get solutions that perform poorly with regard to the objective of our interest. For example, in 3 objective optimization, some solutions are in the corners of the Pareto surface that perform well in two objectives, but poorly in the other. These solutions affect the average performance that we considered in the comparison. In the case of *Leave-one-out* approach, although all three robustness measures are considered, we consider them in two steps. For example, in the case of $(\mathcal{O}-1)_{nER}$, some solutions perform quite well in LCC, but not as well in SG (solutions in one side of the Pareto front). But those solutions are unlikely to be picked by the second step, when we pick the solution that perform best in nER, because of the low correlation between LCC and nER. Hence a solution that performs well in SG is picked by the *Leave-one-out* approach. A similar pattern is seen in Figures 6.3(b) and 6.3(c) as well.

As shown in Figure 6.4, when heuristic edge addition methods are considered, adding edges among the nodes with high degrees results in much better performance in terms of SG. In fact, Rich-Rich performs 2^{nd} best among all the edge addition methods explored here when SG is considered. But Rich-Rich edge addition performs poorly when LCC and nER are considered.

In Table 6.4, we show the average rank of each edge addition method. The values in Table 6.4 are the ranks of all 15 cases (3 robustness measures of interest and 5 percentages of edge addition) for each network. The edge addition methods corresponding to low ranks perform well on all 3 robustness measures of interest. According to the results, adding edges based on the *Leave-one-out* approach yields the best overall robustness in all networks considered in the study.

6.4.2 Network robustness after node attacks

In this section, we investigate how the networks (with robustness improved by edge addition) performed during a phase of multiple node attacks. In this study, we consider two types of node attacks. In targeted node attacks, the nodes with the high degrees (and their

Table 6.4: Average robustness ranks of edge addition methods; smaller values represent greater robustness

Method of edge addition	Average Rank			
	EuroRoad	OpenFlights	US Airports	US Powergrid
SG	5.3	6.1	6.2	6.1
LCC	5.9	5.1	4.7	5.2
nER	5.6	5.3	5.5	5.2
$(\mathcal{O}-1)_{nER}$	**3.3**	3.4	3.5	3.7
$(\mathcal{O}-1)_{LCC}$	**3.3**	**3.3**	3.5	**3.1**
$(\mathcal{O}-1)_{SG}$	3.4	3.4	**3.3**	3.3
$\mathcal{O}_{ave}$	5.1	5.3	5.5	5.1
Rich-Rich	7.9	7.9	8.0	7.6
Poor-Poor	7.7	8.3	7.5	9.3
Rich-Poor	8.9	8.9	9.3	8.7
Random	9.4	8.9	8.9	8.5

corresponding edges) get removed from the network; and in random node attacks, a set of nodes selected randomly from the network (and their incident edges) get removed from the network.

Here, after edge addition to improve the robustness of the network, we attack a set of selected nodes in the network. After the attack, we recalculate the network robustness values. In Table 6.5, we show the robustness values of the OpenFlights network during targeted node attacks and in Table 6.6, we show the robustness values of the same network during random node failures. Suppose as the initial network we consider the network improved by adding 3.3% edges. In the column corresponding to 0% node attacks, we show the robustness value after 3.3% edges have been added to the network. Each latter column refers to robustness values calculated after a certain percentage of nodes are removed from the network.

Prior to any node attacks, the highest value of SG is given by the network to which the edges are added by optimizing SG. In the case of targeted attacks (Table 6.5), as the number of attacked nodes increases, the highest SG values are obtained by the networks to which the edges were added by the $(\mathcal{O}-1)_{nER}$ approach. For the network to which the edges were added by optimizing SG, the SG value reduces sharply as the number of nodes attacked increases. As expected, this sharp decrease is seen in the network to which the edges were added by Rich-Rich approach. In the cases of LCC and nER, for all node attack levels, the highest robustness values are shown by the networks to which the edges were added to optimize LCC and nER respectively. But, even in those two cases, the networks to which the edges were added by the *Leave-one-out* approach show high robustness values even when subjected to targeted node attacks.

Table 6.6 shows the results for the case of random node attacks. In this case, the network created by adding edges to improve SG shows the best SG values during random node attacks. But the robustness of this network is poor when LCC and nER are concerned. A similar behavior is shown by the networks created by adding edges to improve LCC and nER. The networks created by adding edges using *Leave-one-out* approach show high overall robustness when all three measures of robustness are considered. Hence we conclude that the networks created by adding edges using *Leave-one-out* approach retain the high overall robustness during a phase of multiple random node attacks as well.

6.5 Concluding Remarks

When edges are added to a network, the properties of the network change. The amount of change depends on the importance of the set of edges added to the network. In this chapter we addressed the following problem : *Given a network and a budget, how should a set of 'key' edges be selected to be added to the network in order to maximally improve the overall robustness of the network?* Towards this goal, first we discuss the network robustness measures that have been proposed and widely used. Then, we analyze the properties of these robustness measures and identify their similarities and dissimilarities using correlation analysis. Then, we use the *leave-k-out* approach to optimize multiple robustness measures of interest to improve the overall robustness of a network. Experimental evi-

Table 6.5: Robustness values during targeted node attacks - OpenFlights network

Robustness value	Edge addition method	Percentage nodes attacked					
		0%	2%	5%	7.5%	10%	15%
SG	SG	36.292	27.686	14.966	12.632	8.575	4.477
	LCC	21.984	14.811	9.488	7.141	6.314	3.526
	ER	20.976	14.132	9.018	6.614	6.002	3.349
	$(\mathcal{O}-1)_{nER}$	30.391	24.289	12.865	11.478	9.466	5.425
	$(\mathcal{O}-1)_{LCC}$	28.087	17.300	11.035	8.189	7.365	4.223
	$(\mathcal{O}-1)_{SG}$	27.883	17.200	10.952	8.238	7.317	4.625
	Rich-Rich	31.367	23.934	12.757	10.518	7.280	3.727
	Poor-Poor	20.630	13.893	8.905	6.737	5.915	3.280
	Rich-Poor	22.736	15.339	9.627	7.254	6.403	3.531
LCC	SG	0.095	0.087	0.075	0.072	0.066	0.051
	LCC	0.219	0.209	0.200	0.189	0.169	0.148
	ER	0.146	0.132	0.112	0.107	0.094	0.064
	$(\mathcal{O}-1)_{nER}$	0.168	0.168	0.159	0.151	0.134	0.115
	$(\mathcal{O}-1)_{LCC}$	0.161	0.152	0.137	0.127	0.121	0.094
	$(\mathcal{O}-1)_{SG}$	0.167	0.165	0.153	0.143	0.132	0.099
	Rich-Rich	0.092	0.085	0.073	0.071	0.063	0.049
	Poor-Poor	0.154	0.147	0.139	0.136	0.120	0.096
	Rich-Poor	0.093	0.086	0.076	0.075	0.064	0.048
ER	SG	0.0008	0.0007	0.0004	0.0004	0.0001	7.68E-05
	LCC	0.0008	0.0008	0.0005	0.0005	0.0003	0.0002
	ER	0.0016	0.0014	0.0011	0.0010	0.0006	0.0004
	$(\mathcal{O}-1)_{nER}$	0.0012	0.0011	0.0008	0.0007	0.0005	0.0004
	$(\mathcal{O}-1)_{LCC}$	0.0013	0.0013	0.0008	0.0008	0.0004	0.0003
	$(\mathcal{O}-1)_{SG}$	0.0013	0.0012	0.0007	0.0007	0.0004	0.0003
	Rich-Rich	0.0006	0.0005	0.0003	0.0003	2.30E-05	5.78E-06
	Poor-Poor	0.0008	0.0008	0.0004	0.0001	4.01E-06	2.27E-07
	Rich-Poor	0.0006	0.0006	0.0003	0.0003	3.68E-05	5.56E-06

dence shows the improvement in multiple robustness measures when the new edges are added using our algorithm. The key edge identification and addition method proposed in this study improves multiple robustness measures of interest simultaneously, and this can be extremely important in real world applications. For example, when funds need to be allocated to add new roads to a road network, the objectives of interest would include reducing the distance between cities, keeping the cities connected even if some central cities

Table 6.6: Robustness values during random node attacks - OpenFlights network

Robustness value	Edge addition method	Percentage nodes attacked					
		0%	2%	5%	7.5%	10%	15%
SG	SG	36.292	31.046	28.240	27.938	27.021	25.591
	LCC	21.984	20.340	20.131	19.541	17.356	16.931
	ER	20.976	19.333	19.077	18.683	16.777	16.206
	$(\mathcal{O}-1)_{nER}$	30.391	27.124	25.985	24.482	24.072	23.276
	$(\mathcal{O}-1)_{LCC}$	28.087	25.407	24.307	23.971	23.619	21.921
	$(\mathcal{O}-1)_{SG}$	27.883	24.931	24.521	23.897	23.176	21.782
	Rich-Rich	31.367	26.218	24.108	23.361	22.591	21.679
	Poor-Poor	20.630	19.232	19.215	17.994	16.418	16.350
	Rich-Poor	22.736	21.795	21.516	20.316	19.577	18.875
LCC	SG	0.095	0.088	0.087	0.079	0.078	0.076
	LCC	0.219	0.155	0.136	0.131	0.129	0.125
	ER	0.146	0.098	0.095	0.095	0.094	0.092
	$(\mathcal{O}-1)_{nER}$	0.168	0.104	0.102	0.102	0.102	0.096
	$(\mathcal{O}-1)_{LCC}$	0.161	0.099	0.096	0.095	0.098	0.093
	$(\mathcal{O}-1)_{SG}$	0.167	0.114	0.104	0.105	0.103	0.100
	Rich-Rich	0.092	0.090	0.087	0.089	0.086	0.081
	Poor-Poor	0.154	0.115	0.111	0.109	0.116	0.109
	Rich-Poor	0.093	0.089	0.089	0.088	0.083	0.081
ER	SG	0.0008	0.0007	0.0006	0.0006	0.0006	0.0006
	LCC	0.0008	0.0006	0.0006	0.0006	0.0006	0.0005
	ER	0.0016	0.0011	0.0011	0.0011	0.0010	0.0010
	$(\mathcal{O}-1)_{nER}$	0.0012	0.0009	0.0009	0.0008	0.0008	0.0007
	$(\mathcal{O}-1)_{LCC}$	0.0013	0.0010	0.0009	0.0009	0.0008	0.0008
	$(\mathcal{O}-1)_{SG}$	0.0013	0.0009	0.0009	0.0009	0.0008	0.0008
	Rich-Rich	0.0006	0.0005	0.0005	0.0005	0.0004	0.0004
	Poor-Poor	0.0008	0.0006	0.0006	0.0005	0.0005	0.0005
	Rich-Poor	0.0006	0.0005	0.0004	0.0004	0.0004	0.0004

become inaccessible, etc. For such an application, one can select the objectives accordingly and use the edge addition algorithm proposed by this study in order to improve the overall robustness of the underlying system.

CHAPTER 7
CONCLUSIONS AND FUTURE WORK

We summarize the obtained results and give the conclusions of this study, in Section 7.1. Then, in Section 7.2, we present some interesting future research directions.

7.1 Conclusions

Key player identification is an important problem in network science, and earlier studies have mainly focused on the identification of key nodes. In this dissertation, we have investigated the effects of identifying key players which optimize multiple properties of interest, in different well known applications of network science. To identify the sets of key players which optimize multiple properties of interest, we have used a multi-objective optimization algorithm. Our algorithm converts the network into a binary string, and then the sets of key players that optimize multiple objectives are identified by executing a genetic algorithm. In multi-objective optimization problems, a large number of Pareto-optimal solutions (each of which is non-dominated by any other solution) is identified. But decision-makers require one or two ‘good’ solutions to be used in their applications. To accommodate this, we proposed a two step process for multi-objective optimization, viz., *Leave-k-out* approach.

Our results show that the *Leave-k-out* approach is successful in identifying a subset of solutions from the original Pareto set, while reducing the computational time significantly $(p < 0.001)$. Selecting the appropriate k value in the *Leave-k-out* approach is a trade-off between the performance and the computational cost. We have also shown that the sets of key players who possess multiple important characteristics, identified using the proposed *Leave-k-out* approach, outperform the sets of key players identified by the previous approaches in multiple well known applications such as the *Eventual Information Limitation (EIL)* problem and improving the fault tolerance of the *smart grid*.

We also used a multi-objective optimization algorithm to identify key edges in a network. In the context of network robustness, the key edges can be defined as the set of edges that upon addition would improve the robustness of the network the most. The results show that the multi-objective key edges identified using the *Leave-k-out* approach improve the overall robustness of the network the most, compared to the key edges identified using previous approaches.

To improve the running time of the key player identification algorithm, we proposed a sampling algorithm based on the degree centrality of the nodes in the network. We have shown that when the networks are sampled using the degree centrality based sampling approach, most of the key nodes in the original network gets preserved in the sampled network. Our experimental results on the *EIL* problem and the *Immunization* problem show that, in almost 90% of the cases considered, even when the multi-objective key players were identified on the 50% degree centrality sample, these key players perform better than single objective key players identified on the entire network. In addition, degree centrality based sampling reduces the running time of the *Leave-k-out* approach for key player identification significantly.

In conclusion, in multiple real-life applications, viz., *Eventual Information Limitation*, *Immunization*, improving the fault tolerance of the *Smart Grid*, improving network robustness, the key players obtained by multi-objective approach are preferred compared to the key players identified by the previous approaches. As discussed earlier, previous approaches for key player identification consider a single property of interest, and identifies the nodes/edges that maximizes that property of interest. The poor performance of such measures in the aforementioned applications suggest that key players with multiple desirable properties are required for high performance in real-life applications. For a simple example, consider the set of key nodes identified for the Prisoners network by Eigenvector centrality and multi-objective approach (shown in Figure 4.5). Clearly, the set of key nodes identified by Eigenvector centrality is located in one part of the network, and the set of key nodes identified by the multi-objective approach is distributed across the network. If these key nodes were used to initiate the spread of a certain message (similar to viral marketing application) across the network, using the key nodes identified by the multi-objective approach is much more advantageous as they can more directly influence different communities of the network.

7.2 Future Research Directions

Some specific future research directions, which can be explored following the results of this study, are elaborated below.

1. **Identification of multi-objective key players in dynamic social networks:** Dynamicity is an essential characteristic of many real-life networks. For example, as a social network evolve with time, it is common to find people leaving the network, new people joining the network, new connections being formed, and existing connections being deleted. As networks change over time, identification of key players at different points in time is important for some applications.

   To reduce the computational effort, it is important to devise algorithms that can update the set of key players identified in the network dynamically. At the initial time point, the algorithm will identify the set of key players using the current approach for multi-objective key player identification. Then, at each time step of interest, the algorithm will incrementally update the objective values for the already identified key players depending on the changes that occurred in the network during the last time-step. Also, the nodes/edges (that were not part of the initially identified set of key players) that increased their objective values significantly will be considered to be included in the set of key players for the network. This approach avoids recomputation by exploiting information from earlier computations and tracking the changes that occur in the network. Also, to improve the execution time of the initial multi-objective optimization, better binary representations of networks and possibilities of parallel computations can be considered.

2. **Identification of important objectives for different applications:**

   As discussed earlier in this dissertation, key players identified on networks are used in multiple important applications. In this dissertation, we used well known centrality measures as the properties of interest for multiple applications. But, for certain applications, some properties will be more important than others. In such applications, to achieve high performance the algorithm should optimize the more important objectives. This has the advantage that the algorithm's execution time will be shorter. In addition, as the identified set of key players optimizes the most important properties, the performance of the key players should also be high.

   To identify the most important properties for different problems, more extensive experiments can be conducted. One possible approach to identify the set of most important properties is as follows.

   i. Start with a comprehensive set of properties (objectives), the set of key players should possess for the application.

   ii. Use problem specific knowledge to sort the properties in deceasing order of importance.

   iii. Starting with the most important property, add one property at a time to the multi-objective optimization algorithm for key player identification and evaluate the performance of the identified set of key players for the application.

iv. The most important properties (objectives) for the application are the objectives that upon removal reduce the performance of the key players the most.

v. Use the *knee method*[6] to identify the set of most important properties for the application.

3. **Edge sampling for key edge identification:**

   Degree centrality based node sampling was used in this dissertation to reduce the network size by removing the unimportant nodes before key node identification. Similarly, if unimportant edges from the network could be removed before executing the key edge identification, the execution time could be reduced without compromising the quality of the identified key edges. Hence, edge sampling methods which can preserve most of the key edges in the sample will be investigated in the future.

In the big picture, key player identification has considerable potential for cross-domain collaborations in a variety of fields. It would be interesting to study the different properties that the key players exhibit in diverse fields. In addition, as the real-world networks such as social networks continue to grow in size and the technology to collect, represent and store data continue to advance, the amount of data available for analysis increases with time. Hence, developing algorithms that can accommodate the volume, variety, velocity and veracity of data, and interpret large networks present interesting challenges to explore.

References

[1] "Did a cyber attack cause the blackout in turkey?" http://www.dailysabah.com/nation/2015/04/01/did-a-cyber-attack-cause-the-blackout-in-turkey, 2015-04-01 (accessed 2015-04-11). [Online]. Available: http://www.dailysabah.com/nation/2015/04/01/did-a-cyber-attack-cause-the-blackout-in-turkey

[2] M. J. Alenazi and J. P. Sterbenz, "Comprehensive comparison and accuracy of graph metrics in predicting network resilience," *work*, vol. 24, p. 25.

[3] G. Andersson, P. Donalek, R. Farmer, N. Hatziargyriou, I. Kamwa, P. Kundur, N. Martins, J. Paserba, P. Pourbeik, J. Sanchez-Gasca *et al.*, "Causes of the 2003 major grid blackouts in north america and europe, and recommended means to improve system dynamic performance," *Power Systems, IEEE Transactions on*, vol. 20, no. 4, pp. 1922–1928, 2005.

[4] J. E. Baker, "Adaptive selection methods for genetic algorithms," in *Proceedings of an International Conference on Genetic Algorithms and their applications*. Hillsdale, New Jersey, 1985, pp. 101–111.

[5] A.-L. Barabasi and E. Bonabeau, "Scale-free networks," *Scientific American*, 2003.

[6] S. Bechikh, L. B. Said, and K. Ghédira, "Searching for knee regions of the pareto front using mobile reference points," *Soft Computing*, vol. 15, no. 9, pp. 1807–1823, 2011.

[7] P. J. Bentley and J. P. Wakefield, *Finding Acceptable Solutions in the Pareto-Optimal*

Range using Multiobjective Genetic Algorithms. London: Springer London, 1998, pp. 231–240. [Online]. Available: http://dx.doi.org/10.1007/978-1-4471-0427-8_25

[8] A. Bigdeli, A. Tizghadam, and A. Leon-Garcia, “Comparison of network criticality, algebraic connectivity, and other graph metrics,” in *Proceedings of the 1st Annual Workshop on Simplifying Complex Network for Practitioners*. ACM, 2009, p. 4.

[9] T. Blickle and L. Thiele, “A mathematical analysis of tournament selection.” in *ICGA*, 1995, pp. 9–16.

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

[11] P. Boldi and M. Rosa, “Robustness of social and web graphs to node removal,” *Social Network Analysis and Mining*, vol. 3, no. 4, pp. 829–842, 2013.

[12] P. Boldi, M. Rosa, and S. Vigna, *Robustness of social networks: Comparative results based on distance distributions*. Springer, 2011.

[13] P. Bonacich, “Factoring and weighting approaches to status scores and clique identification,” *Journal of Mathematical Sociology*, vol. 2, no. 1, pp. 113–120, 1972.

[14] S. P. Borgatti, “Identifying sets of key players in a social network,” *Computational & Mathematical Organization Theory*, vol. 12, no. 1, pp. 21–34, 2006.

[15] U. Brandes, “A faster algorithm for betweenness centrality*,” *Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[16] M. Broecheler, P. Shakarian, and V. Subrahmanian, “A scalable framework for modeling competitive diffusion in social networks,” in *Social Computing (SocialCom), 2010 IEEE Second International Conference on*. IEEE, 2010, pp. 295–302.

[17] C. Budak, D. Agrawal, and A. El Abbadi, “Limiting the spread of misinformation in social networks,” in *Proceedings of the 20th international conference on World wide web*. ACM, 2011, pp. 665–674.

[18] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin, “Catastrophic cascade of failures in interdependent networks,” *Nature*, vol. 464, no. 7291, pp. 1025–1028, 2010.

[19] D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos, “Epidemic thresholds in real networks,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 4, p. 1, 2008.

[20] H. Chan and L. Akoglu, “Optimizing network robustness by edge rewiring: A general framework,” *Data Mining and Knowledge Discovery (DAMI)*, 2016.

[21] H. Chan, L. Akoglu, and H. Tong, “Make it or break it: Manipulating robustness in large networks,” in *Proceedings of the 2014 SIAM Data Mining Conference*. SIAM, 2014, pp. 325–333.

[22] D. Charles, *The origin of species*. John Murry London, 1929.

[23] P. Chaudhari, R. Dharaskar, and V. Thakare, “Computing the most significant solution from pareto front obtained in multi-objective evolutionary,” *International Journal of Advanced Computer Science and Applications*, vol. 1, no. 4, pp. 63–68, 2010.

[24] M. Cheikh, B. Jarboui, T. Loukil, and P. Siarry, “A method for selecting pareto optimal solutions in multiobjective optimization,” *Journal of Informatics and Mathematical Sciences*, vol. 2, no. 1, pp. 51–62, 2010.

[25] Y. Chen, G. Paul, S. Havlin, F. Liljeros, and H. E. Stanley, “Finding a better immunization strategy,” *Physical review letters*, vol. 101, no. 5, p. 058701, 2008.

[26] C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary algorithms for solving multi-objective problems*. Springer, 2002, vol. 242.

[27] C. A. Coello Coello, "An empirical study of evolutionary techniques for multiobjective optimization in engineering design," 1996.

[28] D. W. Corne and J. D. Knowles, "Techniques for highly multiobjective optimisation: some nondominated points are better than others," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, 2007, pp. 773–780.

[29] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: a survey," *ACM Computing Surveys (CSUR)*, vol. 45, no. 3, p. 35, 2013.

[30] K. A. De Jong, "Analysis of the behavior of a class of genetic adaptive systems," 1975.

[31] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," in *Parallel problem solving from nature PPSN VI*. Springer, 2000, pp. 849–858.

[32] K. Deb and T. Goel, "Controlled elitist non-dominated sorting genetic algorithms for better convergence," in *Evolutionary multi-criterion optimization*. Springer, 2001, pp. 67–81.

[33] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 4, pp. 577–601, 2014.

[34] F. Di Pierro, "Many-objective evolutionary algorithms and applications to water resources engineering," Ph.D. dissertation, University of Exeter, 2006.

[35] N. Drechsler, R. Drechsler, and B. Becker, “Multi-objective optimisation based on relation favour,” in *International conference on evolutionary multi-criterion optimization*. Springer, 2001, pp. 154–166.

[36] N. M. EJ, “Analysis of weighted networks,” *Physical Review E*, vol. 70, no. 5, p. 056131, 2004.

[37] W. Ellens, “Effective resistance and other graph measures for network robustness,” Ph.D. dissertation, Master thesis, Leiden University, 2011. Available at www. math. leidenuniv. nl/scripties/EllensMaster. pdf, 2011.

[38] W. Ellens and R. E. Kooij, “Graph measures and network robustness,” *arXiv preprint arXiv:1311.5064*, 2013.

[39] M. Everett and S. Borgatti, “Extending centrality,” *Models and methods in social network analysis*, vol. 35, no. 1, pp. 57–76, 2005.

[40] M. G. Everett and S. P. Borgatti, “The centrality of groups and classes,” *The Journal of mathematical sociology*, vol. 23, no. 3, pp. 181–201, 1999.

[41] M. Farina and P. Amato, “On the optimal solution definition for many-criteria optimization problems,” in *Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American*, 2002, pp. 233–238.

[42] M. Fiedler, “Algebraic connectivity of graphs,” *Czechoslovak mathematical journal*, vol. 23, no. 2, pp. 298–305, 1973.

[43] R. A. Fisher, *The genetical theory of natural selection: a complete variorum edition*. Oxford University Press, 1930.

[44] C. M. Fonseca, P. J. Fleming *et al.*, “Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization.” in *ICGA*, vol. 93. Citeseer, 1993, pp. 416–423.

[45] L. C. Freeman, “A set of measures of centrality based on betweenness,” *Sociometry*, pp. 35–41, 1977.

[46] M. Garza-Fabre, G. T. Pulido, and C. A. C. Coello, “Ranking methods for many-objective optimization,” in *Mexican International Conference on Artificial Intelligence*. Springer, 2009, pp. 633–645.

[47] A. Ghosh, S. Boyd, and A. Saberi, “Minimizing effective resistance of a graph,” *SIAM review*, vol. 50, no. 1, pp. 37–66, 2008.

[48] M. Girvan and M. E. Newman, “Community structure in social and biological networks,” *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[49] D. E. Goldberg and K. Deb, “A comparative analysis of selection schemes used in genetic algorithms,” *Foundations of genetic algorithms*, vol. 1, pp. 69–93, 1991.

[50] D. E. Goldberg *et al.*, *Genetic algorithms in search optimization and machine learning*. Addison-wesley Reading Menlo Park, 1989, vol. 412.

[51] J. Gómez-Gardeñes and Y. Moreno, “From scale-free to erdos-rényi networks,” *Physical Review E*, vol. 73, no. 5, p. 056124, 2006.

[52] M. S. Granovetter, “The strength of weak ties,” *American journal of sociology*, pp. 1360–1380, 1973.

[53] D. Hadka and P. Reed, “Borg: An auto-adaptive many-objective evolutionary computing framework,” *Evolutionary computation*, vol. 21, no. 2, pp. 231–259, 2013.

[54] J. A. Hartigan and M. A. Wong, “Algorithm as 136: A k-means clustering algorithm,” *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[55] H. J. Herrmann, C. M. Schneider, A. A. Moreira, J. S. Andrade Jr, and S. Havlin, “Onion-like network topology enhances robustness against malicious attacks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2011, no. 01, p. P01027, 2011.

[56] J. Horn, N. Nafpliotis, and D. E. Goldberg, “A niched pareto genetic algorithm for multiobjective optimization,” in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*. Ieee, 1994, pp. 82–87.

[57] Z. Huang, C. Wang, S. Ruj, M. Stojmenovic, and A. Nayak, “Modeling cascading failures in smart power grid using interdependent complex networks and percolation theory,” in *Industrial Electronics and Applications (ICIEA), 2013 8th IEEE Conference on*. IEEE, 2013, pp. 1023–1028.

[58] Z. Huang, C. Wang, M. Stojmenovic, and A. Nayak, “Balancing system survivability and cost of smart grid via modeling cascading failures,” *Emerging Topics in Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 45–56, 2013.

[59] Z. Huang, C. Wang, T. Zhu, and A. Nayak, “Cascading failures in smart grid: Joint effect of load propagation and interdependence,” *Access, IEEE*, vol. 3, pp. 2520–2530, 2015.

[60] M. U. Ilyas and H. Radha, “Identifying influential nodes in online social networks using principal component centrality,” in *Communications (ICC), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–5.

[61] H. Jain and K. Deb, “An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: handling constraints and extending to an adaptive approach,” *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 4, pp. 602–622, 2014.

[62] A. Jamakovic and P. Van Mieghem, *On the robustness of complex networks by using the algebraic connectivity*. Springer, 2008.

[63] W. Jun, M. Barahona, T. Yue-Jin, and D. Hong-Zhong, "Natural connectivity of complex networks," *Chinese Physics Letters*, vol. 27, no. 7, p. 078902, 2010.

[64] C. Kang, C. Molinaro, S. Kraus, Y. Shavitt, and V. Subrahmanian, "Diffusion centrality in social networks," in *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*. IEEE Computer Society, 2012, pp. 558–564.

[65] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[66] L. Katzir, E. Liberty, O. Somekh, and I. A. Cosma, "Estimating sizes of social networks via biased sampling," *Internet Mathematics*, no. just-accepted, 2014.

[67] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 137–146.

[68] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley, and H. A. Makse, "Identification of influential spreaders in complex networks," *Nature Physics*, vol. 6, no. 11, pp. 888–893, 2010.

[69] J. M. Kleinberg, "Hubs, authorities, and communities," *ACM Computing Surveys (CSUR)*, vol. 31, no. 4es, p. 5, 1999.

[70] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety*, vol. 91, no. 9, pp. 992–1007, 2006.

[71] L. T. Le, T. Eliassi-Rad, and H. Tong, “Met: A fast algorithm for minimizing propagation in large graphs with small eigen-gaps,” in *Proceedings of the 2015 SIAM International Conference on Data Mining, SDM*, vol. 15. SIAM, 2015, pp. 694–702.

[72] C.-Y. Lee, “Correlations among centrality measures in complex networks,” *arXiv preprint physics/0605220*, 2006.

[73] S. H. Lee, P.-J. Kim, and H. Jeong, “Statistical properties of sampled networks,” *Physical Review E*, vol. 73, no. 1, p. 016102, 2006.

[74] J. Leskovec and C. Faloutsos, “Sampling from large graphs,” in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 631–636.

[75] F. Linton, “Centrality in social networks conceptual clarification,” *Social networks*, vol. 1, no. 3, pp. 215–239, 1979.

[76] M. Lowe, H. Fan, and G. Gereffi, “US smart grid,” 2011.

[77] D. Lusseau and M. E. Newman, “Identifying the role that animals play in their social networks,” *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 271, no. Suppl 6, pp. S477–S481, 2004.

[78] D. MacRae, “Direct factor analysis of sociometric data,” *Sociometry*, pp. 360–371, 1960.

[79] S. W. Mahfoud, “Crowding and preselection revisited,” *Urbana*, vol. 51, p. 61801, 1992.

[80] F. D. Malliaros, V. Megalooikonomou, and C. Faloutsos, “Fast robustness estimation in large social graphs: Communities and anomaly detection.” in *SDM*, vol. 12. SIAM, 2012, pp. 942–953.

[81] B. Marc, “Betweenness centrality in large complex networks,” *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 163–168, 2004.

[82] M. Marchiori and V. Latora, “Harmony in the small-world,” *Physica A: Statistical Mechanics and its Applications*, vol. 285, no. 3, pp. 539–546, 2000.

[83] N. Mark, “A measure of betweenness centrality based on random walks,” *Social networks*, vol. 27, no. 1, pp. 39–54, 2005.

[84] N. Masuda, “Immunization of networks with community structure,” *New Journal of Physics*, vol. 11, no. 12, p. 123018, 2009.

[85] H. Mühlenbein and D. Schlierkamp-Voosen, “Predictive models for the breeder genetic algorithm i. continuous parameter optimization,” *Evolutionary computation*, vol. 1, no. 1, pp. 25–49, 1993.

[86] K. G. Murty, *Operations research: deterministic optimization models*. Prentice-Hall, Inc., 1994.

[87] S. A. Myers and J. Leskovec, “Clash of the contagions: Cooperation and competition in information diffusion.” in *ICDM*, vol. 12. Citeseer, 2012, pp. 539–548.

[88] M. Newman, *Networks: an introduction*. Oxford University Press, 2010.

[89] M. E. Newman, “Spread of epidemic disease on networks,” *Physical review E*, vol. 66, no. 1, p. 016128, 2002.

[90] T. H. Nguyen, J. Tsai, A. X. Jiang, E. Bowring, R. T. Maheswaran, and M. Tambe, “Security games on social networks.” in *AAAI Fall Symposium: Social Networks and Social Contagion*, 2012.

[91] E. Niiler, “Energy grid: Safe from cyber attack?” http://news.discovery.com/tech/apps/smart-grid-cyber-attacks-110901.htm, 2012-05-09

(accessed 2015-04-11). [Online]. Available: http://news.discovery.com/tech/apps/smart-grid-cyber-attacks-110901.htm

[92] K. Okamoto, W. Chen, and X.-Y. Li, "Ranking of closeness centrality for large-scale social networks," in *Frontiers in Algorithmics*. Springer, 2008, pp. 186–195.

[93] D. Ortiz-Arroyo and D. A. Hussain, "An information theory approach to identify sets of key players," in *Intelligence and Security Informatics*. Springer, 2008, pp. 15–26.

[94] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." 1999.

[95] G.-s. Peng and J. Wu, "Optimal network topology for structural robustness based on natural connectivity," *Physica A: Statistical Mechanics and its Applications*, vol. 443, pp. 212–220, 2016.

[96] B. Phillip, "Power and centrality: A family of measures," *American journal of sociology*, pp. 1170–1182, 1987.

[97] S. D. Reis, Y. Hu, A. Babino, J. S. Andrade Jr, S. Canals, M. Sigman, and H. A. Makse, "Avoiding catastrophic failure in correlated networks of networks," *Nature Physics*, vol. 10, no. 10, pp. 762–767, 2014.

[98] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[99] S. Ruj and A. Pal, "Analyzing cascading failures in smart grids under random and targeted attacks," in *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*. IEEE, 2014, pp. 226–233.

[100] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the 1st international Conference on Genetic Algorithms*. L. Erlbaum Associates Inc., 1985, pp. 93–100.

[101] C. M. Schneider, T. Mihaljev, S. Havlin, and H. J. Herrmann, "Suppressing epidemics with a limited amount of immunization units," *Physical Review E*, vol. 84, no. 6, p. 061911, 2011.

[102] C. M. Schneider, A. A. Moreira, J. S. Andrade, S. Havlin, and H. J. Herrmann, "Mitigation of malicious attacks on networks," *Proceedings of the National Academy of Sciences*, vol. 108, no. 10, pp. 3838–3841, 2011.

[103] C. M. Schneider, N. Yazdani, N. A. Araújo, S. Havlin, and H. J. Herrmann, "Towards designing robust coupled networks," *Scientific reports*, vol. 3, 2013.

[104] B. Shams and M. Khansari, "Immunization of complex networks using stochastic hill-climbing algorithm," in *Computer and Knowledge Engineering (ICCKE), 2013 3th International eConference on*. IEEE, 2013, pp. 283–288.

[105] J. Shao, S. V. Buldyrev, S. Havlin, and H. E. Stanley, "Cascade of failures in coupled network systems with multiple support-dependence relations," *Physical Review E*, vol. 83, no. 3, p. 036116, 2011.

[106] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary computation*, vol. 2, no. 3, pp. 221–248, 1994.

[107] G. Stewart and J.-G. Sun, "Matrix perturbation theory (computer science and scientific computing)," 1990.

[108] S. Sudeng and N. Wattanapongsakorn, "Post pareto-optimal pruning algorithm for multiple objective optimization using specific extended angle dominance," *Engineering Applications of Artificial Intelligence*, vol. 38, pp. 221–236, 2015.

[109] A. Sydney, C. Scoglio, and D. Gruenbacher, “Optimizing algebraic connectivity by edge rewiring,” *Applied Mathematics and computation*, vol. 219, no. 10, pp. 5465–5479, 2013.

[110] T. Tanizawa, S. Havlin, and H. E. Stanley, “Robustness of onionlike correlated networks against targeted attacks,” *Physical Review E*, vol. 85, no. 4, p. 046109, 2012.

[111] A. Tizghadam and A. Leon-Garcia, “Autonomic traffic engineering for network robustness,” *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 1, pp. 39–50, 2010.

[112] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos, “Gelling, and melting, large graphs by edge manipulation,” in *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2012, pp. 245–254.

[113] H. Tong, B. A. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau, “On the vulnerability of large graphs,” in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 1091–1096.

[114] J. Tsai, T. H. Nguyen, and M. Tambe, “Security games for controlling contagion.” in *AAAI*, 2012.

[115] T. W. Valente, K. Coronges, C. Lakon, and E. Costenbader, “How correlated are network centrality measures?” *Connections (Toronto, Ont.)*, vol. 28, no. 1, p. 16, 2008.

[116] H. Wang and P. Van Mieghem, “Algebraic connectivity optimization via link addition,” in *Proceedings of the 3rd International Conference on Bio-Inspired Models of Network, Information and Computing Sytems*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, p. 22.

[117] T. Watanabe and N. Masuda, “Enhancing the spectral gap of networks by node removal,” *Physical Review E*, vol. 82, no. 4, p. 046102, 2010.

[118] D. J. Watts and S. H. Strogatz, “Collective dynamics of âĂŸsmall-worldâĂŹnetworks,” *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[119] B. Wiegmans, “Gridkit: European and north-american high-voltage power grid extracts,” 2016. [Online]. Available: http://dx.doi.org/10.5281/zenodo.47317

[120] L. J. Wismans, T. Brands, E. C. Van Berkum, and M. C. Bliemer, “Pruning and ranking the pareto optimal set, application for the dynamic multi-objective network design problem,” *Journal of advanced transportation*, vol. 48, no. 6, pp. 588–607, 2014.

[121] Z.-X. Wu and P. Holme, “Onion structure and network robustness,” *Physical Review E*, vol. 84, no. 2, p. 026106, 2011.

[122] A. Yazdani, R. A. Otoo, and P. Jeffrey, “Resilience enhancing expansion strategies for water distribution systems: A network theory approach,” *Environmental Modelling & Software*, vol. 26, no. 12, pp. 1574–1582, 2011.

[123] S. Yoon, S. Lee, S.-H. Yook, and Y. Kim, “Statistical properties of sampled networks by random walks,” *Physical Review E*, vol. 75, no. 4, p. 046114, 2007.

[124] A. Zeng and W. Liu, “Enhancing network robustness against malicious attacks,” *Physical Review E*, vol. 85, no. 6, p. 066130, 2012.

[125] Q. Zhang and H. Li, “Moea/d: A multiobjective evolutionary algorithm based on decomposition,” *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 6, pp. 712–731, 2007.

[126] E. Zitzler, M. Laumanns, L. Thiele, E. Zitzler, E. Zitzler, L. Thiele, and L. Thiele, “Spea2: Improving the strength pareto evolutionary algorithm,” 2001.

[127] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach,” *evolutionary computation, IEEE transactions on*, vol. 3, no. 4, pp. 257–271, 1999.

VITA

NAME OF THE AUTHOR: R. Chulaka Gunasekara

PLACE OF BIRTH: Colombo, Sri Lanka

SCHOOLS ATTENDED:

- University of Moratuwa, Department of Computer Science and Engineering, Moratuwa, Sri Lanka (2005-2009)
- Syracuse University, Department of Electrical Engineering and Computer Science, Syracuse, New York, USA (2011-2016)

DEGREES AWARDED:

- B.Sc. Engineering (Hons.) in Computer Science and Engineering (1st Class Honors), University of Moratuwa, Sri Lanka (2009)
- Master of Science in Computer Science, Syracuse University, Syracuse, New York, USA (2015)

PUBLICATIONS:

- R Chulaka Gunasekara, Kishan Mehrotra, Chilukuri K Mohan, "Improving the Robustness of the Smart Grid using a Multi-Objective Key Player Identification Approach", International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2016), San Francisco, CA, USA, August 2016
- Ashok Kumar, R. Chulaka Gunasekara, Kishan G. Mehrotra and Chilukuri K. Mohan, "Algorithms for Fast Estimation of Social Network Centrality Measures", International Journal of Big Data Intelligence, May 2016 (Submitted)

- R Chulaka Gunasekara, Kishan Mehrotra, Chilukuri K Mohan, "Multi-objective optimization to identify key players in large social networks", Social Network Analysis and Mining 5 (1), 1-20, 2015
- R Chulaka Gunasekara, Kishan Mehrotra, Chilukuri K Mohan, "Multi-Objective Optimization to Identify Key Players in Social Networks", International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014), Beijing, China, August 2014
- R Chulaka Gunasekara, Kishan Mehrotra, Chilukuri K Mohan, "Multi-Objective Restructuring in Social Networks", International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013), Niagara Falls, Canada, August 2013
- R Chulaka Gunasekara, Kishan Mehrotra, Chilukuri K Mohan, "Robustness Measures for Social Networks", International Conference on Computer Science and Information Technology (ICSIT 2012), Hyderabad, India, August 2012
- R. Chulaka Gunasekara, Akila Geethal, Mafaz Hassan, Tharindu Mathew and Shehan Perera, "Prophetia: Artificial Intelligence for TravelBox Technology", Advances in Computational Intelligence, pp 21-34, Springer 2009
- P.P. Wijegunawardana, K.T.Buddhika, D.P. Jayamanne, R. Chulaka Gunasekera, "DecisionAI: A Framework to Automate the Decision Making Process", 6th International Conference on Information and Automation for Sustainability (ICIAfS 2012), Beijing, China, September 2012
- C. N. Joseph, S. Kokulakumaran, K. Srijeyanthan, A.Thusyanthan, R. Chulaka Gunasekara, and C. D. Gamage, "A Framework for Whole-Body Gesture Recognition from Video Feeds", 5th International Conference on Industrial and Information Systems (ICIIS 2010), Karnataka, India, July 2010

PROFESSIONAL EXPERIENCE:

- Summer Research Intern - IBM Watson Group, Astor Place, New York, USA (May 2015 - August 2015)
- Summer Research Intern - IBM Research, Yorktown Heights, New York, USA (May 2014 - August 2014)
- Trainee Software Engineer - Direct FN Ltd, Colombo, Sri Lanka (October 2007 - May 2008)

ACADEMIC EXPERIENCE:

- Graduate Research Assistant - Syracuse University (August 2015 - Present)
- Graduate Teaching Assistant - Syracuse University (August 2011 - May 2015)

ASSOCIATION MEMBERSHIPS:

- Student Member: ACM, IEEE
- Sun Certified Java Professional (2008)