# Optimal Resource Allocation in

# Social and Critical Infrastructure Networks

by

Anisha Mazumder

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved October 2016 by the
Graduate Supervisory Committee:

Arunabha Sen, Chair
Andrea Richa
Guoliang Xue
Martin Reisslein

ARIZONA STATE UNIVERSITY

December 2016

ABSTRACT

We live in a networked world with a multitude of networks, such as communication networks, electric power grid, transportation networks and water distribution networks, all around us. In addition to such physical (infrastructure) networks, recent years have seen tremendous proliferation of social networks, such as Facebook, Twitter, LinkedIn, Instagram, Google+ and others. These powerful social networks are not only used for harnessing revenue from the infrastructure networks, but are also increasingly being used as "non-conventional sensors" for monitoring the infrastructure networks. Accordingly, nowadays, analyses of social and infrastructure networks go hand-in-hand. This dissertation studies resource allocation problems encountered in this set of diverse, heterogeneous, and interdependent networks. Three problems studied in this dissertation are encountered in the physical network domain while the three other problems studied are encountered in the social network domain.

The first problem from the infrastructure network domain relates to distributed files storage scheme with a goal of enhancing robustness of data storage by making it tolerant against large scale geographically-correlated failures. The second problem relates to placement of relay nodes in a deployment area with multiple sensor nodes with a goal of augmenting connectivity of the resulting network, while staying within the budget specifying the maximum number of relay nodes that can be deployed. The third problem studied in this dissertation relates to complex interdependencies that exist between infrastructure networks, such as power grid and communication network. The progressive recovery problem in an interdependent network is studied whose goal is to maximize system utility over the time when recovery process of failed entities takes place in a sequential manner.

The three problems studied from the social network domain relate to influence propagation in adversarial environment and political sentiment assessment in various states in a country with a goal of creation of a "political heat map" of the country. In the first problem of the influence

propagation domain, the goal of the second player is to restrict the influence of the first player, while in the second problem the goal of the second player is to have a larger market share with least amount of initial investment.

# DEDICATION

To Ma (Ratna Mazumder) and Baba (Narayan Chandra Mazumder)

ACKNOWLEDGMENTS

I would like to express my sincerest and deepest appreciation and gratitude to my advisor Dr. Arunabha Sen for his unwavering support, incessant help as well as his continuous faith in me. He has essentially built my research capabilities and inspired me through his insights, guidance, ideas, kindness, and foresight. I would also like to thank Dr. Andrea Richa, Dr. Guoliang Xue and Dr. Matrin Reisslein for their constant help and support and for serving as my brilliant Ph.D. committee.

I am also deeply thankful to my seniors in our lab - Dr. Sujogya Banerjee for his solid support, advices and motivations specially during the initial months of my doctoral study, as well as Dr. Shahrzad Shirazipourazad for being a role model whom I have always tried to follow. I am also greatly thankful to my labmates and co-authors Chenyang and Arun for their continuous help - it was truly wonderful working with them. I am also deeply thankful to all my friends in Kolkata (my home town) and Tempe for their continued support and motivation. In particular, I am sincerely thankful to Arindam for his enormous support throughout my endeavor of the doctoral program.

Finally, I am truly indebted to my parents and my sister and brother-in-law for their infinite support and limitless encouragement through each day of my doctoral studies. I would have never been able to achieve the doctorate degree had it not been for them.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

# Chapter 1

# INTRODUCTION

Ours is a networked world. Critical infrastructures ranging from power grids to transportation systems, water distribution systems to satellite systems to the Internet - all are networked. These critical infrastructure systems are of extreme importance both in our personal day-to-day life as well as in national economy, security, public health and safety. Furthermore, we, as individuals, are also networked. This makes the social networks of people, both inside and outside of a country, impact national economy, security, public health and safety. Additionally, the last few years have seen tremendous proliferation of online social networks, such as Facebook, Twitter, LinkedIn, Instagram, Nextdoor, Google+ and others. These social networks are not only effectively used for generating higher revenue from the critical infrastructure networks, but they are also being increasingly used as "non-conventional sensors" for monitoring the health and security of the critical infrastructure networks. As a result, in present times, there is an increasing effort for combining analyses of social and infrastructure networks. In particular, analysis for prudent and judicious allocation of resources in these diversified networks is critical for optimal performance. This requires in-depth understanding of the structure and function of the networks and available networking resources. This dissertation focuses on these varied, heterogeneous, intertwined and interdependent networks, and studies a variety of problems in these domains for effective resource utilization.

## 1.1 Motivation and Challenges

The information technology (IT) sector forms a backbone of a country's communication, economy, and security. Cloud is one of the most ubiquitous commodity used by both public and government of a nation. This dissertation starts with a distributed data storage problem in the IT critical infrastructure networks. Distributed storage of data files in different nodes of an IT storage network enhances its fault tolerance capability by offering protection against node and link failures. Reliability is often achieved through redundancy in one of the following two ways: (i) storage of multiple copies of the entire file at different locations (nodes), or (ii) storage of file segments (not entire files) at different node locations. In the $(\mathcal{N}, \mathcal{K})$ file distribution scheme, $\mathcal{N}$ file segments from a file $F$ are created in such a way that it is possible to reconstruct the entire file, just by accessing any $\mathcal{K} \leq \mathcal{N}$ segments. For the reconstruction scheme to work, it is essential that the $\mathcal{K}$ segments of the file are stored in nodes that are *connected* in the network. However, in the event of node/link failures, the network might become disconnected (*i.e.*, split into several connected components). We focus on node failures that are *spatially-correlated* or *region-based*. Such failures are often encountered in disaster situations or natural calamities where only the nodes in the disaster zone are affected. Let there be an *allocated budget* on the amount of allowed storage. Consider the objective of designing a file storage scheme to ensure that no matter which region is destroyed, resulting in fragmentation of the network, a *largest connected component* of the residual network will have enough file segments with which to *reconstruct the entire file*. In case the least cost to achieve this objective is within the *allocated budget*, the storage design will be *all region fault-tolerant*. In case the least cost *exceeds the allocated budget*, design of an all region fault-tolerant file storage system is impossible. The first goal of this research is to design file storage schemes that will be *maximum region fault-tolerant within the allocated budget*. The second goal

of this research is to investigate the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on storage requirements for ensuring *all region* fault-tolerant design. In [1], the authors study the design of all region fault-tolerant system in a general network and they have shown the problem to be NP-hard. So, the third goal of this research is to investigate the problem of designing an all region fault-tolerant system for a mesh network. Mesh networks have a specific structure and so the problem of designing an all region fault-tolerant network in case of mesh networks may not be NP-hard as in the case of a general network.

In order to ensure availability and survivability of critical infrastructure, sensors are deployed to monitor the health of the critical infrastructure networks. The intention is to detect anomalies and undertake quick response such that further degradation can be avoided. Connectivity of such wireless sensor networks (WSNs) is critical to ensure that the information gathered by the sensors is not lost and can be fully utilized. To achieve connectivity in the WSNs relay nodes are deployed. The relay node placement problem in the wireless sensor network have been studied extensively in the last few years. The goal of most of these problems is to place the fewest number of relay nodes in the deployment area so that the network formed by the sensors nodes and the relay nodes is *connected*. Most of these studies are conducted for the unconstrained budget scenario, in the sense that there is an underlying assumption that no matter however many relay nodes are needed to make the network connected, they can be procured and deployed. However, in a fixed budget scenario, the expenses involved in procuring the minimum number of relay nodes to make the network connected may exceed the budget. Although in this scenario, one has to give up the idea of having a network connecting all the sensor nodes, one would still like to have a network with high level of "connectedness". This dissertation investigates the possible techniques of achieving maximal "connectedness" in a WSN under a budget on the number of available relay nodes.

There has been an increasing recognition among network operators as well as the research community that critical infrastructure networks do not operate in isolation and in fact they are highly interdependent. For instance, the power grid entities, such as the SCADA systems that control power stations and sub-stations, receive their commands through communication networks. On the other hand, the entities of the communication network, such as routers and base stations, cannot operate without electric power. Cascading failures in the power grid, are even more complex now because of the coupling between power grid and communication network. Due to this coupling, not only entities in power networks, such as generators and transmission lines, can trigger power failure, communication network entities, such as routers and optical fiber lines, can also trigger failure in power grid. Thus it is essential that the interdependency between different types of networks be understood well. This will ensure that preventive measures can be taken to avoid cascading catastrophic failures in multi-layered network environments and also restorative measures can be taken in the event of some unavoidable failure. Accordingly, a number of models have been proposed to analyze interdependent networks in recent years. However, most of the models are unable to capture the complex interdependencies that exist between these networks. Utilizing a recently proposed model, this dissertation provides techniques for progressive recovery from failure. The goal of the progressive recovery problem is to maximize the system utility over the entire duration of the recovery process.

Online Social Networks (OSNs) provide an enormous volume of extremely rich data for analyzing human sentiment about people, places, events, political, economic and religious activities. It is becoming increasingly clear that analysis of such data can provide great insights on the social, economic, political and cultural aspects of the participants of these networks, and even some of the non participants. As part of the US Department of Defense sponsored Minerva project, in this dissertation, a large volume of Twitter data has been analyzed to understand

radical political activity in the provinces of Indonesia. Based on machine learning and probabilistic analysis of radical or counter radical sentiments expressed in tweets by Twitter users, Heat Maps of Indonesia have been generated. These Heat Maps visually demonstrate the degree of radical activities in various provinces of Indonesia. The resultant Heat Maps can be effectively used for monitoring radical activities in a country and deployment of resources for ensuring peace and stability. Also, if radical or counter radical activities were to propagate from one region to another, these Heat Maps would be able to visually demonstrate the same. The findings of this Twitter data analysis was effectively validated by one of the most respected political think tanks of Indonesia.

In the domain of social networks, it has been observed that the decision of an individual regarding adoption of a product or technology is, more often than not, heavily influenced by their friends and family. Numerous models of propagation of influence have been proposed in the past several years. In the real world, there are different competing products and innovations that try to garner as many loyal followers as possible. Over the past several years, there has been a significant interest in the economics, computer science and sociology research community to study different variations of social network problems in a competitive environment. Such problems often focus on identification of a set of key individuals in a given social network by competing interests (players) that try to attain their own individual objectives. This dissertation studies two such problems in a two player adversarial setting. As the first problem, this dissertation studies the scenario in which given a weighted social network graph where the first player has identified her set of initial adopters, the goal for the second player is to select a subset of nodes of minimum cumulative weight with which to contain the reachability of the first player to less than half the total weight of the nodes of the graph. This problem is also relevant in the fields of damage control of epidemiology, disaster control etc. In the second problem, an unweighted social network graph is given along with a set of nodes already selected by the

first player as her set of initial adopters. With this knowledge, the second player tries to select the minimum number of nodes such that at the end of the influence propagation, the second player wins over more individuals in the social network compared to the first player.

## 1.2 Contributions

The contributions of this dissertation can be categorized as follows:

- Critical Infrastructure Networks research:
    1. region-based fault-tolerant design of distributed data storage networks
    2. relay node placement problem under budget constraint to ensure maximal "connectedness"
    3. progressive recovery from failure in multi-layered interdependent network
- Social Network Analysis research:
    1. generation of heat map of spatio-temporal signal of a geographical area
    2. containment of influence in social networks in adversarial setting
    3. influence propagation in social networks in adversarial setting

### 1.2.1 Region-Based Fault-tolerant Distributed File Storage System Design in Networks

This dissertation presents in-depth study on design of region-based fault-tolerant file storage system in distributed data storage networks. In this dissertation, contributions in this domain are made in three different aspects:

- Maximum region fault-tolerant system design: A budget constrained distributed file system design problem is introduced and this research provides solutions that maximizes

the number of regions that can be made fault-tolerant, within the specified budget. An approximation algorithm for the problem is provided. The performance of the approximation algorithm is evaluated through simulation on two real networks. The simulation results demonstrate that the worst case experimental performance is significantly better than the worst case theoretical bound. Moreover, the approximation algorithm almost always produce near optimal solution in a fraction of time needed to find the optimal solution.

- Impact of coding parameters: This dissertation presents analytical results demonstrating that the choice of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ may have significant impact on the storage that will be necessary to achieve reliability.
- All region fault-tolerant system design in mesh networks: A polynomial time algorithm for optimal storage allocation to achieve all region fault-tolerant design in a mesh network is presented and extensive experimentation is conducted to evaluate the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirement to provide all region fault tolerance with varying size of the mesh and the fault region.

### 1.2.2 Budget Constrained Relay Node Placement Problem for Maximal "Connectedness"

This dissertation studies the relay node placement problem under budget constraint. To this end, this research introduces two metrics for measuring "connectedness" of a disconnected graph and studies the problem whose goal is to design a network with *maximal* "connectedness", subject to a fixed budget constraint. It is shown that both versions of the problem are NP-complete and heuristics for their solution are provided. The problem is shown to be *non-trivial* even when the number of sensor nodes is as few as *three*. The performance of the heuristics is evaluated through simulation.

### 1.2.3 Progressive Recovery from Failure in Multi-layered Interdependent Network Using a New Model of Interdependency

This dissertation provides detailed study on progressive recovery from failure in a multi-layered interdependent network (such as power and communication networks) in the light of a recently proposed model of interdependency. In this dissertation, it has been shown that the problem can be solved in polynomial time in some special case, whereas for some others, the problem is NP-complete. For the former case, an optimal polynomial time algorithm has been proposed. For the latter cases, when the problem is NP-complete, this research provides two approximation algorithms with performance bounds of $2$ and $4$ respectively. Finally, for the most general case (which is certainly NP-complete), an Integer Linear Programming formulation and a heuristic have been proposed. The efficacy of the heuristic is evaluated with both synthetic and real data collected from Phoenix metropolitan area. The experiments show that the heuristic almost always produces near optimal solution.

### 1.2.4 Spatio-Temporal Signal Recovery from Political Tweets in Indonesia

In this dissertation, as a part of the US Department of Defense sponsored Minerva project which was underway in Arizona State University, a large volume of Twitter data has been analyzed through the application of machine learning and probabilistic analysis techniques. The goal of the analysis has been to understand radical political activity in the provinces of Indonesia. In this dissertation, a generic technique is developed for recovering signals pertaining to any country or geographical area and is applied to the Indonesian Twitter dataset. Based on analysis of radical/counter radical sentiments expressed in tweets by Twitter users, the outcome of this research is to create a Heat Map of Indonesia which visually demonstrates

the degree of radical activities in various provinces of Indonesia. The Heat Map of Indonesia is created by computing (i) the *Radicalization Index* and (ii) the *Location Index* of each Twitter user from Indonesia, who has expressed some radical sentiment in her tweets. The conclusions derived from this research matches significantly with the analysis of Wahid Institute, a leading political think tank of Indonesia, thus validating our results.

### 1.2.5 On Social Network Firewall Selection

This dissertation introduces the *w*eighted Segregating Vertex Set (wSVS) problem, in which given a weighted undirected graph with a subset of nodes identified as the seed set of the first player, the goal for the second player is to identify a subset of nodes (firewall) of minimum cumulative weight, such that the total weight of the nodes reachable by the first player is strictly less than the total weight of the nodes not reachable by the first player. Thus, the second player tries to contain the *reach* of the first player within the social network community. This dissertation proves that this problem is NP-complete and provides an optimal solution through the use of Mixed Integer Linear Programming. A heuristic solution for the wSVS problem is also provided and its efficacy is shown through detailed experimentation. The heuristic solution delivers near optimal solution in lesser time compared to that needed to find the optimal solution.

### 1.2.6 Winning with Minimum Investment under Separated Threshold Model (WMI-LT)

Influence propagation problem in social networks under adversarial setting has been studied in this dissertation. In particular, in the problem studied in this dissertation, the goal of the second player is to win over more individuals in the social network compared to the

first player by employing minimum investment for the purpose of incentivization under the Separated Threshold model of influence propagation. The problem is first proven to be NP-Complete and then an $O(log(n))$ approximation algorithm is presented. This is followed by experimental results on synthetic and real world data showing the efficacy of the approximation algorithm. Also, an equivalent random process for the Separated Threshold Model is presented - this equivalent random process facilitates analysis under the Separated Threshold model of influence propagation.

Several segments of the research work that is presented in this dissertation, have been already published in international conferences and journals as listed below:

- **Anisha Mazumder**, Chenyang Zhou, Arun Das, and Arunabha Sen. 2016. "Budget Constrained Relay Node Placement Problem for Maximal "Connectedness"." In IEEE International Conference for Military Communications (MILCOM).
- **Anisha Mazumder**, and Arunabha Sen. 2016. "On Social Network Firewall Selection." In IEEE International Conference on Computing, Networking and Communications (ICNC), Social Computing and Semantic Data Mining.
- Arunabha Sen, **Anisha Mazumder**, Sujogya Banerjee, Arun Das, Chenyang Zhou, and Shahrzad Shirazipourazad. 2015. "Region-based fault-tolerant distributed file storage system design in networks." Networks 66 (4): 380–395.
- **Anisha Mazumder**, Arun Das, Chenyang Zhou, and Arunabha Sen. 2014. "Region based fault-tolerant distributed file storage system design under budget constraint." In IEEE 6th International Workshop on Reliable Networks Design and Modeling (RNDM), 61–68.
- **Anisha Mazumder**, Chenyang Zhou, Arun Das, and Arunabha Sen. 2014. "Progressive Recovery from Failure in Multi-layered Interdependent Network Using a New Model of

Interdependency." In 9th International Conference on Critical Information Infrastructures Security.

- **Anisha Mazumder**, Arun Das, Nyunsu Kim, Sedat Gokalp, Arunabha Sen, and Hasan Davulcu. 2013. "Spatio-temporal signal recovery from political tweets in Indonesia." In IEEE International Conference on Social Computing (SocialCom), 280– 287.

## 1.3 Dissertation Outline

The rest of the dissertation is organized as follows: In Chapter 2, the design of *maximum region based fault-tolerant* distributed file storage networks has been studied. Also, the effect of coding parameters on storage requirements for ensuring *all region* fault-tolerant design is investigated and an optimal algorithm for ensuring *all region* fault-tolerant design in a mesh network is presented. In Chapter 3, the relay node placement problem under budget constraint using two different metrics has been studied. In Chapter 4, the problem of maximizing the system utility over the entire duration of the recovery process in a multi-layered interdependent network is studied using a recently proposed model of interdependency. In Chapter 5, the problem of generating Heat Maps of a geographical region, such as a country, demonstrating the intensity of signals, such as radical activities, has been studied. Chapters 6 and 7 study two problems in the domain of social network analysis. First, Chapter 6 studies the problem of identification of nodes in a social network that can be used to contain the spread of an opposing influence to less than half of the population. Second, in Chapter 7, the identification of influential nodes in social networks in adversarial setting has been discussed. Finally, Chapter 8 concludes the dissertation and presents future direction of this research.

Chapter 2

# REGION-BASED FAULT-TOLERANT DISTRIBUTED FILE STORAGE SYSTEM DESIGN IN NETWORKS

Distributed storage of data files across the nodes of a network enhances fault tolerance and security, and reduces the file retrieval cost. One of the simplest file distribution schemes across the nodes is replication. In this scheme, a data file $F$ of size $|F|$ is replicated $l$ times and a copy of the file is stored in $l$ different nodes in the network. Although this file replication scheme can tolerate failure of up to $l-1$ nodes, it has two major shortcomings: (i) the total storage space used by this scheme over the network is $l \times |F|$, and (ii) if only one node storing the replicated file is compromised, an unauthorized user can have access to the entire file. In order to avoid these shortcomings, and enhance fault tolerance, security and load balancing capability, error-correcting codes have been used extensively in data storage systems and server clusters - such as RAID [2] and DPSS [3]. One well-known scheme for this purpose is the use of $(\mathcal{N}, \mathcal{K})$ erasure codes [4]–[7]. In the $(\mathcal{N}, \mathcal{K})$ *maximum distance separable* (MDS) erasure code based storage system, $\mathcal{N}$ coded segments are created from the original file $F$ and stored in $\mathcal{N}$ nodes of the network (one coded segment per node). The advantage of this storage scheme is that the original file $F$ can be reconstructed by any user in the network just by retrieving and then decoding any $\mathcal{K}$ out of the $\mathcal{N}$ segments. Clearly, location of the storage nodes within the network will have an impact on the ease of retrieval of the segments. The focus of our research is to design a robust file distribution scheme that takes into account the network topology - particularly in the scenario when one or more of the network nodes may be unavailable due to failure.

In a network spanning a large geographical area, the faulty nodes may be *spatially-correlated* (*i.e.*, confined to a *region*). Such failures are often encountered in disaster situations, either natural (earthquake, forest fire, flood or hurricane) or man-made (EMP attack or an enemy bomb in a battle field), where only the nodes/links in the disaster zone are affected. These faults are generally referred to as *spatially-correlated faults* or *region-based faults* [8]. We consider a scenario where failure of nodes (capable of storing data segments) in a single region might disconnect the network. $(\mathcal{N}, \mathcal{K})$ codes ensure that as long as $\mathcal{K}$ file segments survive, the file can be reconstructed. However, this condition alone is insufficient for successful file reconstruction where the network may fragment into two or more connected components and, although more than $\mathcal{K}$ segments survive, none of the components have more than $\mathcal{K} - 1$ segments. To be able to reconstruct the file, one has to ensure that at least one of the connected components has at least $\mathcal{K}$ segments. In a recent paper [1], the authors have proposed $(\mathcal{N}, \mathcal{K})$ coding based distributed file storage scheme which ensures that no matter which region of the network fails, at least one of the largest connected components (LCC) of the resulting fragmented network will have sufficient number of distinct file segments ($\mathcal{K}$) with which to reconstruct the entire file. Since this data storage scheme ensures reconstruction of the original file no matter which region of the network fails, we refer to this as *all region fault-tolerant* (ARFT) design, or the scheme that provides *100% fault coverage*. However, in order to provide 100% coverage against region-based faults, the file segments may have to be stored in a large number of network nodes. Accordingly, storage cost may be quite substantial and may even exceed the allocated *budget*. As such, in a *budget constrained environment* it may not be possible to design such an all-region fault-tolerant distributed storage system. One goal of this research is to design file storage schemes that will be *maximum region fault-tolerant* (MRFT) within the *allocated budget*. In the dissertation, we refer to this problem as the *Budget Constrained Data Distribution Problem* (BCDDP). Our goal of minimization of storage

requirement is driven by the fact that although storage has become less expensive in recent times, the cost of storing large data files (of the order of petabytes, exabytes or higher) is still significantly high.

Although quite a few studies [9]–[11] have proposed the use of $(\mathcal{N}, \mathcal{K})$ coding for reliable distributed file system design, to the best of our knowledge, no one has investigated the impact of the choice of the coding parameters $(\mathcal{N}, \mathcal{K})$ on the storage that will be necessary to provide an *all region fault-tolerant* (ARFT) distributed file system. As mentioned previously, by *all region fault-tolerant system* we mean a system that has the data segments stored in different network nodes in such a way that no matter which region of the network fails, a *largest connected component* of the *residual network* will have $\mathcal{K}$ segments with which to reconstruct the entire file. In this dissertation, we present analytical results demonstrating that the choice of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ may have significant impact on storage that will be necessary to achieve fault tolerance capability.

[1] proves that all region fault tolerant (ARFT) design is NP-complete for general networks. But it is possible to solve the ARFT design problem optimally in polynomial time for networks with specific structure with a specific definition of region. In this dissertation, we present a polynomial time algorithm for optimal storage allocation in a mesh network with (i) specified size of the mesh, (ii) specified size of the fault region, and (iii) coding parameters $\mathcal{N}$ and $\mathcal{K}$. We conduct extensive experimentation to evaluate the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirement to provide *all region fault tolerance* with varying size of the mesh and the fault region.

The contributions in this chapter are as follows:

- We analyze the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirements.
- We present a polynomial time algorithm for the ARFT design problem in a mesh network

with (i) specified size of the mesh, (ii) specified size of the fault region, and (iii) coding parameters $\mathcal{N}$ and $\mathcal{K}$.

- We study the BCDDP and provide a file distribution scheme utilizing $(\mathcal{N}, \mathcal{K})$ coding that ensures that the maximum number of regions are made fault-tolerant within the specified budget.
- We provide optimal solution using Integer Linear Programming and approximate solution with a guaranteed performance bound for the BCDDP.
- We present experimentation to evaluate the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirement to provide *all region fault tolerance* with varying size of the mesh and the fault region.
- We provide experimental evaluation of the performance of the approximation algorithm through simulation using two real networks.

The rest of the chapter is organized as follows. In Section 2.1, we provide two motivating examples for the data distribution problem; in Section 2.2, we discuss prior research work on related topics; In Section 2.3, we provide formulation of the BCDDP; In Section 2.4, we discuss the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirements; In Section 2.5, we present the optimal algorithm for the ARFT design problem in a mesh network; In Section 2.6, we discuss computational complexity for the BCDDP problem; In Section 2.7, we present optimal and approximate solutions for the BCDDP; In Section 2.8, we present our extensive experimentation to evaluate the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirement to provide all region fault tolerance with varying size of the mesh and the fault region; In Section 2.9 we present experimental results for the BCDDP using two real backbone networks respectively.

## 2.1 Motivating Examples

In this section, we provide two examples to demonstrate the effectiveness of the $(\mathcal{N}, \mathcal{K})$ coding scheme for file storage and the distinction between the DDP and the BCDDP. We use the term "robust" to imply that the distribution scheme enables the non-faulty nodes of a largest connected component to reconstruct the entire file after a region-based fault strikes the network. Next, we also provide a motivating example for the BCDDP. It may be noted that a region may be defined in terms of the *topology* or the *geometry* (*i.e.*, the layout of the network in the two-dimensional plane) of the network. An example of a region in terms of topology could be a subgraph with a specified diameter $d$. An example of a region in terms of geometry could be a circular area of radius $r$.

The examples shown in Fig. 1(a) and 1(b) demonstrate the data distribution schemes with and without coding. Both schemes are robust against region-based faults, where a region is defined to be a subgraph of diameter one. The regions are shown as circular disks in Fig. 1. It is assumed that the storage capacity of each node is one. Fig. 1 shows the distribution of three uncoded file segments $A, B$ and $C$ of a file $F$. As shown in Fig. 1(a), the uncoded segments $A, B$ and $C$ must be stored in at least six nodes of the network, in order to make it robust against a region-based fault. However, using $(\mathcal{N}, \mathcal{K})$ coding, the same result can be achieved by storing data segments in at most four nodes of the network, when ($\mathcal{N} = 4$ and $\mathcal{K} = 3$. This can be accomplished by creating a new coded segment $(A \bigoplus B \bigoplus C)$ where $\bigoplus$ represents the XOR operation, and storing this segment along with segments $A, B$ and $C$. This is shown in Fig. 1(b).

Fig. 2 shows an $(\mathcal{N}, \mathcal{K})$ coding based data distribution scheme of the European fiber backbone network of a major network service provider [12], in the presence of a massive region-based fault. The fault is assumed to be a circular area of radius $150$ miles. In this example,

$\mathcal{N} = 20$ and $\mathcal{K} = 10$. The colored nodes (also marked with tags such as $col_1, col_2$ and so on for denoting the different colors) in Fig. 2 are the nodes where a coded segment is stored. Nodes with the same color store the same coded segment. Each node is assumed to have a storage capacity of one. The distribution shown in Fig. 2 stores data segments in twenty-two locations (hence, the total storage requirement, denoted by $\sigma$, is twenty-two). In this example twenty-two is the fewest number of locations where the segments have to be stored to ensure robustness against any circular region-based fault of radius $r = 150$ miles, *i.e.*, a largest connected component of the network will have at least $\mathcal{K}$ nodes with distinctly coded file segments.

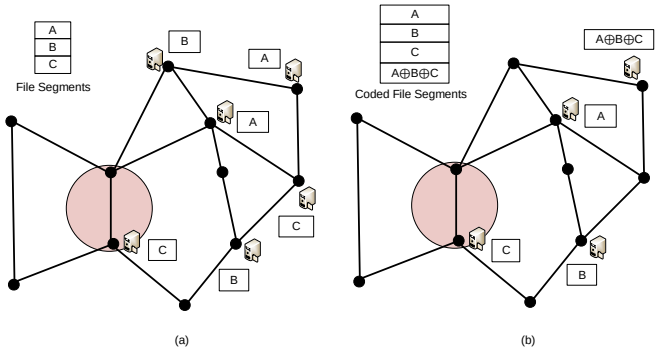



Figure 1: A data storage network with a region-based fault. Fig. (a) uses no coding scheme and uses 6 storage units. Fig. (b) uses an ($\mathcal{N}$, $\mathcal{K}$) coding scheme and uses only 4 storage units.

We elaborate on the motivation of the BCDDP with the help of an example. Fig. 3 shows a data storage network with a region-based fault and the corresponding *largest connected component* (LCC). In order to make this network ARFT (*i.e.*, to provide 100% fault coverage), with the size of the circular fault region shown in Fig. 3(a), and the coding parameters $\mathcal{N}$ = 4 and $\mathcal{K}$ = 2, the encoded file segments must be stored in at least four different nodes.

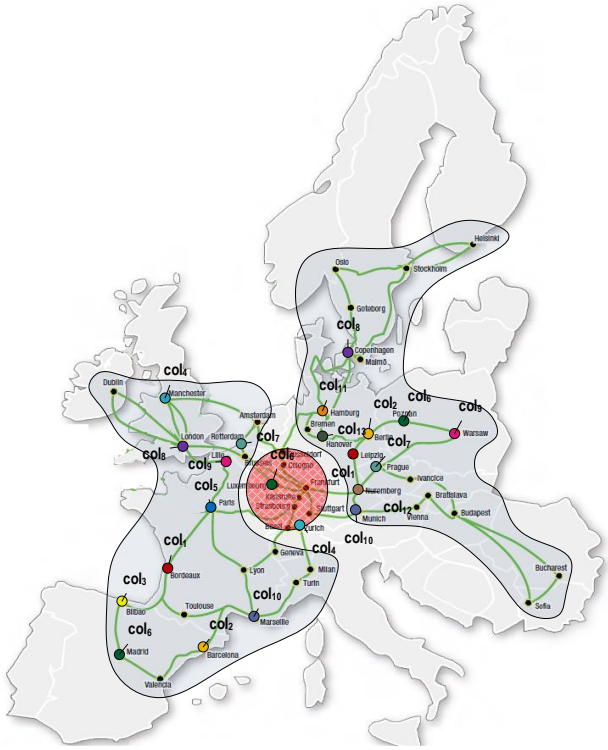
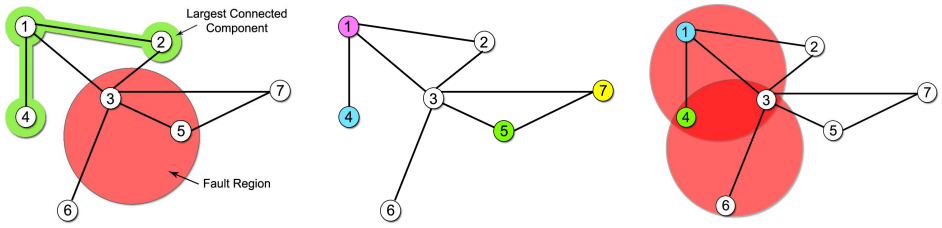



Figure 2: Fiber backbone of a major Europe network provider with a region-fault and distribution of file segments.

A possible solution is shown in Fig. 3(b), where the segments are stored in nodes 1, 4, 5 and 7. However, if due to the budget constraint, we are allowed to store encoded file segments in only two nodes, 100% coverage can no longer be provided. A possible solution where the segments are stored in only two locations (nodes 1 and 4) is shown in Fig. 3(c). Some of the regions that will remain vulnerable in this situation are also shown in Fig. 3(c). Table 1 lists all the twenty-one regions for this network, where a region is a circular area of size shown in Fig. 3(a). The largest connected components (LCCs) corresponding to each of the regions is also shown in Table 1. In this example, when the fault region encompasses nodes 3 and 5, the network is fragmented into three connected components {1, 2, 4}, {6}, {7}, with the component {1, 2,




(a) (b) (c)

Figure 3: (a) A data storage network with a region-based fault and the corresponding LCC. This network with this size of the circular fault region requires that segments be stored in four different locations to provide 100% coverage, (b) The locations where the segments can be stored to provide 100% coverage, (c) A possible solution when (due to the budget constraint) segments can be stored only in two locations, and some of the unprotected or vulnerable regions.

4} being the largest. When the budget is reduced from four to two, only thirteen out of the twenty-one regions can be covered with the lower budget. Storing the file segments in nodes 1 and 4 makes the data distribution scheme maximum region fault-tolerant under the reduced budget.

In the table, the completely covered regions are marked with a tick. It may be noted that in this example, when the budget is reduced from four to two, the fault region coverage drops from 100% to 61%. The goal of the BCDDP is to maximize coverage subject to the budget constraint.

## 2.2 Related Work

The networking research community over the last decade has seen a heightened level of interest in *spatially-correlated* or *region-based* faults in networks [8], [13]–[19]. Although all these studies delve into some aspects of robustness, none of them focus on the robust and optimal data storage problem in the presence of region-based faults.

| Regions | LCC | Regions | LCC |
|---|---|---|---|
| {1} | {2,3,5,6,7} | {4,6} | {1,2,3,5,7} |
| {2} | ✓ {1,3,4,5,6,7} | {1,3} | {5,7} |
| {3} | ✓ {1,2,4} | {2,5} | ✓ {1,3,4,6} |
| {4} | {1,2,3,5,6,7} | {3,4} | {5,7} |
| {5} | ✓ {1,2,3,4,6,7} | {3,6} | ✓ {1,2,4} |
| {6} | ✓ {1,2,3,4,5,7} | {5,7} | ✓ {1,2,3,4,6} |
| {7} | ✓ {1,2,3,4,5,6} | {1,3,4} | {5,7} |
| {3,5} | ✓ {1,2,4} | {3,4,6} | {5,7} |
| {2,3} | ✓ {1,4} | {2,3,5} | ✓ {1,4} |
| {1,4} | {2,3,5,6,7} | {2,5,7} | ✓ {1,3,4,6} |
| {2,7} | ✓ {1,3,4,5,6} | | |

Table 1: Regions and the corresponding LCC.

Error-correcting codes have been used extensively in enhancing the performance, reliability and fault tolerance capability in data storage systems [4], [6], [7], [9]–[11], [20]–[22]. In [4], [6], Dimakis *et al.* consider node failures, and use erasure codes to solve the repair problem of a node (*i.e.*, how to replace the data on a failed node). However, the problem under study in this dissertation is considerably different from the one in [4], [6]. While [4], [6] focus on the repair problem and study the trade-off between storage and the bandwidth requirement, this study is directed towards development of a robust scheme that allows reconstruction of the file after a few nodes disappear due to a region-based failure. Moreover, our technique accounts for the topology of the network while designing the data distribution scheme.

Distribution of coded file segments among different nodes in the network, taking the topology of the network into account, has been studied in [5], [9]–[11], [21], [22]. The focus of this line of research is in developing a file distribution scheme so that each node in the network can recreate the original file by accessing $\mathcal{K}$ file segments from nodes within their $r$ hop neighborhoods. Jiang *et al.* solve this problem optimally for networks with special structure such as trees [10] and tori [21]. This chapter of the dissertation is along the same line

as that of [10] and [9] and our objective is also to minimize the total storage in the network. However, there exists a major distinction between our research and that of [10] and [9]. In [9], [10], [22], the authors solve the problem when there is no fault in the network, whereas this dissertation discusses the file segment distribution scheme on a network of arbitrary topology considering a region-based fault in the network.

In [5], Jiang *et al.* solve the file segment distribution problem with the goal of minimizing $\mathcal{N}$, subject to the constraint that any node $v$ in the network will have at least $\mathcal{K}$ distinct file segments within a specified radius $r$. However, minimizing $\mathcal{N}$ does not necessarily minimize the amount of storage $\sigma$ that will be necessary to satisfy the proximity requirement. The example in Fig. 4 shows that $\sigma$ is $5$ when the goal is to minimize $\mathcal{N}$ and $\sigma$ is $4$ when the goal is to minimize total storage (while satisfying the proximity requirement that the entire file be reconstructible by collecting $\mathcal{K}$ segments within one hop neighbors). In this example, it is assumed that each node can store at most one file segment. This example shows a tradeoff between $\mathcal{N}$ and $\sigma$. The storage requirement decreases with increase in $\mathcal{N}$.

In [1], the authors have proposed $(\mathcal{N}, \mathcal{K})$ coding based distributed file storage scheme which ensures that no matter which region of the network fails, at least one of the largest connected components (LCC) of the resulting fragmented network will have sufficient number of distinct file segments ($\mathcal{K}$) with which to reconstruct the entire file. Since this data storage scheme ensures reconstruction of the original file no matter which region of the network fails, we refer to this as *all region fault-tolerant* (ARFT) design, or the scheme that provides *100% fault coverage*. However, in order to provide 100% coverage against region-based faults, the file segments may have to be stored in a large number of network nodes. Accordingly, storage cost may be quite substantial and may even exceed the allocated *budget*. As such, in a *budget constrained environment* it may not be possible to design such an all-region fault-tolerant distributed storage system.

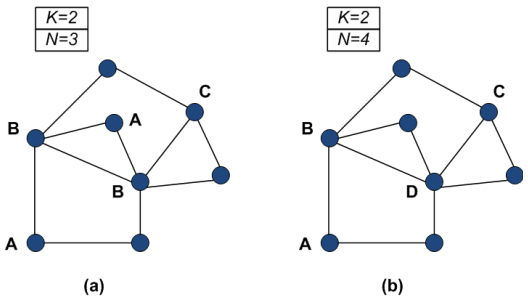



Figure 4: Example showing the tradeoff between the value of $\mathcal{N}$ and the storage $\sigma$ required in a network. Total file segments available in Fig. (a) are $\{A, B, C\}$, so storage required is $5$ while file segments available in Fig. (b) are $\{A, B, C, D\}$, so storage required is $4$.

## 2.3 Problem Formulation

In this section, we present formal statement of the BCDDP. It may be noted that in this dissertation, we focus on region-based faults in a geometric setting. A file $F$ of length $\mathcal{M}$ is split into $\mathcal{K}$ segments each of length $\mathcal{M}/\mathcal{K}$. Using erasure coding techniques like maximum distance separable (MDS) codes, Reed-Solomon (RS) codes, etc., these $\mathcal{K}$ segments can be encoded to produce $\mathcal{N}$ coded segments ($\mathcal{N} \geq \mathcal{K}$). According to the erasure coding theory, retrieving any $\mathcal{K}$ out of $\mathcal{N}$ segments is sufficient to reconstruct the entire file $F$.

The network is represented by a graph $G = (V, E)$ where $V$ is the set of $n$ nodes in the network and $E$ is the set of links between them. Each node has a storage capacity for storing the file segments. For the file $F$, a node $v_i \in V$ can store maximum of $\alpha_i$ file segments. Nodes in the network are assumed to have both storage and processing units, so any node can recreate the file $F$ by downloading and decoding $\mathcal{K}$ coded segments from other nodes. Let the layout of $G$ on a two-dimensional plane be $LG = (Pt, L)$ where $Pt = \{pt_1, \ldots, pt_n\}$ and $L = \{l_1, \ldots, l_m\}$ are the sets of points (representing nodes) and straight lines (representing links) respectively. We consider a region to be a circular area $R$ of radius $r$ on this plane. Although with this definition of a region, there could potentially be an infinite number of regions in the network deployment area, we need to consider only the *distinct* regions. Two regions are

considered *distinct* if they do not include the same set of nodes and edges. It has been shown in [8] and [23] that the number of distinct regions in wireless and wired networks are $O(n^2)$ and $O(n^4)$ respectively, where $n$ is the number of nodes in the network. We assume that a fault in a region destroys all the nodes and links in that region. Due to a region-based fault, the residual network might get fragmented or disconnected. The *connected components* of a graph are the equivalence classes of nodes under the *"is reachable from"* relation [24]. A connected component of the largest size is a largest connected component (LCC) of the graph. Let $C_i$ be the set of nodes in a largest connected component in network $G$ after region $R_i$ fails, where $i = 1, \ldots, p$. It may be noted that destruction of two distinct regions may result in the same largest connected component (LCC) and as such the number of distinct LCCs may be less than the number of distinct regions. Let $\mathcal{R} = \{R_1, \ldots, R_p\}$ denote the $p$ distinct regions of the network and let the set $\mathcal{C} = \{C_1, \ldots, C_p\}$ represent the set of the $p$ largest connected components of the network $G = (V, E)$ after failure of region $R_i, 1 \leq i \leq p$ (*i.e.*, the nodes and links in the region $R_i$ are removed from the graph $G = (V, E)$). Let $Col = \{col_1, col_2, \ldots, col_{\mathcal{N}}\} = \{1, 2, \ldots, \mathcal{N}\}$ be a set of $\mathcal{N}$ distinct colors. The BCDDP can be formulated as a color assignment problem where each color represents a file segment and assigning a color $col_j,\ 1 \leq j \leq \mathcal{N}$, to a node $v_i$ implies that file segment $j$ is stored in node $v_i$. If the storage capacity of node $v_i$ is $\alpha_i$, then at most $\alpha_i$ colors can be assigned to node $v_i$. The terms color capacity and storage capacity of a node can be used interchangeably. Furthermore, in this dissertation, we have assumed that $\alpha_i = 1$ for each $v_i \in V$. If the storage budget is $B$, the goal of the BCDDP is to store $B$ encoded segments (colors) in the nodes of the network in such a way that the largest number of LCCs have at least $\mathcal{K}$ distinct file segments. We assume $B \geq \mathcal{K} \geq 1$. Thus, the BCDDP ensures that the largest number of regions that can be made fault-tolerant within the budget constraint $B$, is in fact made fault-tolerant.

Formally, the BCDDP can be stated as follows:

*Budget Constrained Data Distribution Problem (BCDDP)*

*INSTANCE:* Given

(i) a graph $G = (V, E)$ where $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$ are the sets of nodes and edges respectively,

(ii) the layout of $G$ on a two-dimensional plane $LG = (Pt, L)$ where $Pt = \{pt_1, \ldots, pt_n\}$ and $L = \{l_1, \ldots, l_m\}$ are the sets of points and lines on the two-dimensional plane,

(iii) a maximum of $\alpha_i = 1$ colors can be assigned to a node $v_i$, $i.e.$, at most $\alpha_i = 1$ file segments can be stored in node $v_i$,

(iv) each region $R$ is a circular area of radius $r$,

(v) a set $Col = \{1, 2, \ldots, \mathcal{N}\}$ of distinct file segments (colors) and the coding parameter $\mathcal{K}$,

(vi) storage budget $B$, which is the maximum number of nodes that can be selected for storing one encoded data segment (equivalently, the maximum number of nodes that can be selected for assignment of colors),

(vii) region-based fault-tolerance parameter $\tau$.

QUESTION: Is there a way to assign colors to a subset $V' \subseteq V, |V'| \leq B$ such that the number of LCCs that receive at least $\mathcal{K}$ distinct colors is at least $\tau$?

It may be noted that a low value of $\mathcal{N}$ implies low decoding complexity. But, a low value of $\mathcal{N}$ also increases the total storage required in the network. In fact, if the value of $\mathcal{N}$ is chosen below a certain threshold then the amount of storage required can increase exponentially. We provide a formal proof of this assertion in Section 2.4. The budget constraint $B$ of the BCDDP ensures that no more than $B$ nodes can be selected for storing encoded data segments. Since the choice of a large value of $\mathcal{N}$ usually leads to a small storage requirement and our goal in this dissertation is to minimize the storage requirement, we have preferred to have as large a value for $\mathcal{N}$ as possible. Now, for the BCDDP, we assume that each node can store

only one encoded data segment, *i.e.*, $\alpha_i = 1$ for each node $v_i \in V$. Since the storage capacity of each node is one, no more than $n$ segments can be stored in the network. Accordingly, we take $\mathcal{N} = n$ in this dissertation.

## 2.4 Impact of the Coding Parameters $\mathcal{N}$ and $\mathcal{K}$ on the Required Storage $\sigma$

In this section, we analyze the impact of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on the storage requirements for the unbudgeted version of the BCDPP. This unbudgeted version of the problem has been studied in [1] and has been called the DDP - but the impact of the coding parameters have not been studied in [1] and is hence studied in this dissertation. For completeness, we next provide the formal statement of the DDP.

Data Distribution Problem (*DDP*)

*INSTANCE:* Given

(i) a graph $G = (V, E)$ where $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$ are the sets of nodes and links respectively,

(ii) the layout of $G$ on a two-dimensional plane $LG = (Pt, L)$ where $Pt = \{pt_1, \ldots, pt_n\}$ and $L = \{l_1, \ldots, l_m\}$ are the sets of points and lines on the two-dimensional plane,

(iii) a maximum of $\alpha_i$ colors can be assigned to a node $v_i$, $i.e.$, at most $\alpha_i$ file segments can be stored in node $v_i$,

(iv) each region $R$ is a circular area of radius $r$,

(v) a set $Col = \{1, 2, \ldots, \mathcal{N}\}$ of distinct colors and the coding parameter $\mathcal{K}$,

(vi) storage parameter $\phi$.

*QUESTION*: Is there a way to assign at most $\mathcal{N}$ colors to the nodes in $V$ such that (i) $\rho_i \leq \alpha_i$, where $\rho_i$ is the number of colors assigned to node $v_i$, (ii) for each possible circular region of

radius $r$, at least one LCC has at least $\mathcal{K}$ distinct colors and (iii) $\sigma = \sum_{i=1}^{n} \rho_i$ is less than or equal to $\phi$?

Since storing each file segment involves a cost, one of the objectives for the DDP is to minimize the total storage used in the whole network. Thus, we note that whereas the goal of the DDP is to make the network *all region fault-tolerant* with *least cost*, the goal of the BCDDP is to make the network *maximum region fault-tolerant* within the *budget constraint*. Also, it may be noted that for the DDP, $\alpha_i \geq 1$.

For analyzing the impact of the coding parameters, we first provide the definitions of two relevant problems, followed by an example that shows that the choice of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ can have considerable impact on the minimum storage requirement $\sigma$. We then prove a theorem to formally establish how significant this difference can be. We note that if $\alpha_i = 1$, the DDP in an abstract form can be represented as the following problem:

*Set Coloring Problem* (SCP): Given a set of elements $S = \{s_1, \ldots, s_n\}$, another set $\mathcal{S} = \{S_1, \ldots, S_p\}$, where $S_i \subseteq S, 1 \leq i \leq p$, integers $\mathcal{K}$ and $\mathcal{N}$, find the smallest subset $S' \subseteq S$, such that the elements of $S'$ can be colored in such a way that each $S_i, 1 \leq i \leq p$, receives at least $\mathcal{K}$ distinct colors and the number of colors used does not exceed $\mathcal{N}$. (Note: not every element in $S_i \subseteq S, 1 \leq i \leq p$, has to be colored.)

If we take $\mathcal{N} \geq n$, the SCP reduces to the following problem.

*Hitting $\mathcal{K}$-Set Problem*: Given a set of elements $S = \{s_1, \ldots, s_n\}$, another set $\mathcal{S} = \{S_1, \ldots, S_t\}$, where $S_i \subseteq S, 1 \leq i \leq t$, and the integer $\mathcal{K}$, find the smallest subset $S_H \subseteq S$, such that for each set $S_i, 1 \leq i \leq t$, $|S_H \cap S_i| \geq \mathcal{K}$.

It may be noted that instances of the *Hitting $\mathcal{K}$-Set problem* are a subset of the instances of the *Set Coloring problem*. Suppose that for a given instance $I_{SCP}$ of the *Set Coloring Problem*, the size of the smallest *Hitting $\mathcal{K}$-Set* of this instance is $\beta$, *i.e.*, $|S_H| = \beta$. This implies

that minimizing the total storage is equivalent to minimizing the number of nodes where one segment of data has to be stored.

*Example 1:* Consider an instance of SCP where $S = \{1, 2, \ldots, 20\}$ and the subsets $\mathcal{S}$=$\{S_1, S_2, ..., S_{28}\}$, $S_i \subseteq S$, $1 \leq i \leq 28$, are shown in Table 2. The solution to SCP is the smallest $S' \subseteq S$ such that the elements of $S'$ can be colored in a way that each $S_i, 1 \leq i \leq t$, receives at least $\mathcal{K} = 2$ distinct colors and the number of colors used does not exceed $\mathcal{N} = 8, 4, 2$. The cardinality of the smallest $S'$, for $\mathcal{N} = 8, 4$, and $2$, are $8$ (where elements $1$ through $8$ of the set $S$ are assigned a color, in other words file segments are placed in nodes $1$ through $8$), $12$ (where elements $1$ through $8$, and $10, 13, 16,$ and $18$ of the set $S$ are assigned a color) and $20$ (where elements $1$ through $20$ of the set $S$ are assigned a color) respectively. It may be noted that the cardinality of $S'$ represents the number of nodes where one segment of the data file has to be stored. The colors assigned to the nodes are represented by letters $A, B, ...$ in Table 2.

In the following discussion we establish a theorem that formalizes our observations from the previous example, that there can be a significant difference in $\sigma$ depending on the choice of $\mathcal{N}$ and $\mathcal{K}$.

**Lemma 2.1.** *In the SCP problem, if $\mathcal{N} \geq \beta$, then $\sigma$ is at most $\beta$.*

*Proof:* If $\mathcal{N} \geq \beta$, then we first solve the *Hitting $\mathcal{K}$-Set* problem on the instance of SCP problem and assign each of the $\beta$ nodes of the solution set $S_H$ a distinct color. It is easy to check that this will satisfy the *coloring constraint* for the SCP problem as each subset $S_i, 1 \leq i \leq t$, will have $\mathcal{K}$ different colors. Thus, the number of nodes that need to be colored in this case, and hence the storage requirement $\sigma$, is at most $\beta$.

**Lemma 2.2.** *In the SCP problem if $\mathcal{N} < \beta$, then $\sigma$ can be as large as $2^\beta$.*

*Proof:* Let us consider an instance of SCP such that $\mathcal{K} = 2$, $n = 2^{\beta}$. W.l.o.g. we assume $s_i = i$ where $s_i \in S$ and $1 \leq i \leq n$. Again w.l.o.g, we assume that the first $\beta$ elements of the set $S$ constitute the solution of the *Hitting $\mathcal{K}$-Set problem*, *i.e.*, $S_H = \{1, 2, \ldots, \beta\}$. The subsets of SCP are as follows:

$\mathcal{S} = \{(i, j, k) | (i, j) \subset \{1, 2, \ldots, \beta\}, k \in \{\beta + 1, \beta + 2, \ldots, n\}\}$. It may be noted that $S_H$ is the solution of the *Hitting $\mathcal{K}$-Set problem* for this instance. All the nodes of $S_H$ must be assigned a color to satisfy the coloring constraint. Since $\mathcal{N} < \beta$, at least one pair of nodes in $S_H$ will have the same color. Suppose that nodes $i$ and $j$ are assigned the same color. Because of this assignment the following collection of subsets will violate the coloring constraint (*i.e.*, each subset must have at least two distinct colors)

$$\{i, j, \beta + 1\}, \{i, j, \beta + 2\}, \ldots, \{i, j, n\}$$

In order to satisfy the coloring constraint, not only nodes $1, \ldots, \beta$ need to be colored, nodes $\beta + 1, \beta + 2, \ldots, n$ must also be assigned a color. Since only one data segment is stored in a node and in order to satisfy the coloring constraint each node must receive a color, the storage requirement $\sigma$ will be equal to $n$. Since we assumed that $n = 2^{\beta}$, the number of nodes that has be colored can be as large as $2^{\beta}$.

**Theorem 2.3.** *The number of nodes that need to be colored to satisfy the coloring constraint $\sigma$ could be as small as $\beta$ or as large as $2^{\beta}$, depending on whether $\mathcal{N} \geq \beta$ or $\mathcal{N} < \beta$ respectively.*

*Proof:* It follows from Lemmas 2.1 and 2.2.

| | $\mathcal{K}=2$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\mathcal{N}=8$ | | | $\mathcal{N}=4$ | | | $\mathcal{N}=2$ | | |
| Color | A | B | | A | B | | A | B | |
| Subset $S_1$ | 1 | 2 | | 1 | 2 | | 1 | 2 | |
| Color | A | C | | A | C | | A | A | B |
| Subset $S_2$ | 1 | 3 | 9 | 1 | 3 | 9 | 1 | 3 | 9 |
| Color | A | D | | A | D | | A | B | |
| Subset $S_3$ | 1 | 4 | | 1 | 4 | | 1 | 4 | |
| Color | A | E | | A | A | B | A | A | B |
| Subset $S_4$ | 1 | 5 | 10 | 1 | 5 | 10 | 1 | 5 | 10 |
| Color | A | F | | A | B | | A | B | |
| Subset $S_5$ | 1 | 6 | | 1 | 6 | | 1 | 6 | |
| Color | A | G | | A | C | | A | A | B |
| Subset $S_6$ | 1 | 7 | 11 | 1 | 7 | 11 | 1 | 7 | 11 |
| Color | A | H | | A | D | | A | B | |
| Subset $S_7$ | 1 | 8 | | 1 | 8 | | 1 | 8 | |
| Color | B | C | | B | C | | B | A | |
| Subset $S_8$ | 2 | 3 | | 2 | 3 | | 2 | 3 | |
| Color | B | D | | B | D | | B | B | A |
| Subset $S_9$ | 2 | 4 | 12 | 2 | 4 | 12 | 2 | 4 | 12 |
| Color | B | E | | B | A | | B | A | |
| Subset $S_{10}$ | 2 | 5 | | 2 | 5 | | 2 | 5 | |
| Color | B | F | | B | B | A | B | B | A |
| Subset $S_{11}$ | 2 | 6 | 13 | 2 | 6 | 13 | 2 | 6 | 13 |

| Color | B | G | | B | C | | B | A | |
|---|---|---|---|---|---|---|---|---|---|
| Subset $S_{12}$ | 2 | 7 | | 2 | 7 | | 2 | 7 | |
| Color | B | H | | B | D | | B | B | A |
| Subset $S_{13}$ | 2 | 8 | 14 | 2 | 8 | 14 | 2 | 8 | 14 |
| Color | C | D | | C | D | | A | B | |
| Subset $S_{14}$ | 3 | 4 | | 3 | 4 | | 3 | 4 | |
| Color | C | E | | C | A | | A | A | B |
| Subset $S_{15}$ | 3 | 5 | 15 | 3 | 5 | 15 | 3 | 5 | 15 |
| Color | C | F | | C | B | | A | B | |
| Subset $S_{16}$ | 3 | 6 | | 3 | 6 | | 3 | 6 | |
| Color | C | G | | C | C | B | A | A | B |
| Subset $S_{17}$ | 3 | 7 | 16 | 3 | 7 | 16 | 3 | 7 | 16 |
| Color | C | H | | C | D | | A | B | |
| Subset $S_{18}$ | 3 | 8 | | 3 | 8 | | 3 | 8 | |
| Color | D | E | | D | A | | B | A | |
| Subset $S_{19}$ | 4 | 5 | | 4 | 5 | | 4 | 5 | |
| Color | D | F | | D | B | | B | B | A |
| Subset $S_{20}$ | 4 | 6 | 17 | 4 | 6 | 17 | 4 | 6 | 17 |
| Color | D | G | | D | C | | B | A | |
| Subset $S_{21}$ | 4 | 7 | | 4 | 7 | | 4 | 7 | |
| Color | D | H | | D | D | A | B | B | A |
| Subset $S_{22}$ | 4 | 8 | 18 | 4 | 8 | 18 | 4 | 8 | 18 |
| Color | E | F | | A | B | | A | B | |
| Subset $S_{23}$ | 5 | 6 | | 5 | 6 | | 5 | 6 | |

| Color | E | G | | A | C | | A | A | B |
|---|---|---|---|---|---|---|---|---|---|
| Subset $S_{24}$ | 5 | 7 | 19 | 5 | 7 | 19 | 5 | 7 | 19 |
| Color | E | H | | A | D | | A | B | |
| Subset $S_{25}$ | 5 | 8 | | 5 | 8 | | 5 | 8 | |
| Color | F | G | | B | C | | B | A | |
| Subset $S_{26}$ | 6 | 7 | | 6 | 7 | | 6 | 7 | |
| Color | F | H | | B | D | | B | B | A |
| Subset $S_{27}$ | 6 | 8 | 20 | 6 | 8 | 20 | 6 | 8 | 20 |
| Color | G | H | | C | D | | A | B | |
| Subset $S_{28}$ | 7 | 8 | | 7 | 8 | | 7 | 8 | |

Table 2: Set coloring with different values of $\mathcal{N}$.

## 2.5 Optimal Algorithm for Data Distribution in a Mesh Network

In [1] it was proved that for arbitrary networks DDP is an NP-complete problem. But it is possible to solve DDP optimally in polynomial time for networks with specific structure with a specific definition of region. In this section we present a polynomial time optimal algorithm for solving DDP in a regular two dimensional grid network of size $(n \times n)$ where failures of the nodes are assumed to be confined to a region defined by a smaller grid of size $(r \times r)$ with $r < n$ and $n$ being a multiple of $r$. In an $(n \times n)$ two dimensional grid network node $u_{ij}$ is located at the position $(i, j)$, $\forall 1 \leq i \leq n, 1 \leq j \leq n$, on the grid (Fig. 5). Faults are assumed to be confined within a smaller grid of size $(r \times r)$ completely contained within the grid network. The fault region can be anywhere inside the grid network, and it is assumed all the nodes in the fault region becomes non operational. The storage capacity of each node is

assumed to be 1 unit. One important property of grid networks is that the residual network will always remain connected even after any region fault. Given a value of $(\mathcal{N}, \mathcal{K})$, Algorithm 1 (DDG) shows a technique to assign colors to minimum number of nodes in this network such that for any region fault, the residual network (which is also the largest connected component) will have at least $\mathcal{K}$ different colors.

The algorithm DDG starts by assigning color 1 to node $u_{1,1}$. The two inner *for-loops* (step 10-31) ensures that colors assigned to nodes in each iteration do not fall within the same region. In each iterations the two outer *for-loops* (step 7-33) appropriately shifts $x$ and $y$ indices of the new node to be colored so that in that iteration maximum number of uncolored nodes can be colored. During the coloring process $r'$ keeps track of the number of nodes colored so far which falls within a single region. The goal is to distribute the colored nodes in such a way that $r'$ is minimum. Each of these outer iterations increases $r'$ by one. If $\mathcal{N} \geq \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$, the algorithm terminates when number of nodes colored $\rho$ is equal to $\mathcal{K} + r'$.

If $\mathcal{N} < \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$ the algorithm will have to reuse some of the already assigned colors to color new nodes. The algorithm reuses colors in such a way that nodes with same color do not fall in the same region (step 14-18). The algorithm then terminates when $\rho = 2\mathcal{K} - (\mathcal{N} - \mathcal{K})(\lceil \frac{n}{r} \rceil^2 - 2)$. In worst case the algorithm assigns color to each node. The time complexity of the algorithm is $O(|U|)$ i.e., $O(n^2)$.

**Theorem 2.4.** *Given a value of $\mathcal{N}$, $\mathcal{K}$ Algorithm 1 (DDG) gives optimal solution.*

*Proof:* Here, we assume that $n$ is a multiple of $r$. First of all, when $\mathcal{K} > n^2 - r^2$ or $\mathcal{N} < \mathcal{K}$ then no solution is feasible. Step 3 of DDG takes care of this infeasibility condition. We find the optimal solution for two different cases:

**CASE I ( $\mathcal{N} \geq \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$ ):** Assume that in the optimal solution nodes are assigned colors in such a way that the maximum number of colored nodes falling within one region is given by

**Algorithm 1:** Data Distribution on Grid Network (DDG)

**INPUT:**

1. The network $G = (U, E)$ with a two dimensional grid topology of size $n \times n$ where node $u_{ij} \in U$,
2. Maximum 1 color can be assigned to $u_{i,j}$, $\forall u_{i,j} \in U$,
3. Color set $\mathcal{C} = \{i, \ldots, \mathcal{N}\}$,
4. Region grid length $r$,
5. Parameter $\mathcal{K}$.

**OUTPUT:** Assignment of colors $c_{i,j} \in \mathcal{C}$ to each node $u_{i,j}$, $\forall 1 \leq i \leq n$ such that for any region faults of size $r \times r$ the residual network $G'$ has at least $\mathcal{K}$ distinct colors and number of nodes colored $\rho$ is minimum;

**if** $\mathcal{K} > n^2 - r^2$ *OR* $\mathcal{N} < \mathcal{K}$ **then**
    No feasible solution exists;

$c = 0$, $\rho = 0$, $r' = 0$;
**for** $l = 0 \rightarrow 2r - 1$ **do**
    **for** $i = \lceil \frac{l+1}{r} \rceil - 1 \rightarrow l$ **do**
        $j = l - i$; $r' = r' + 1$;
        **for** $p = 0 \rightarrow \lceil \frac{n}{r} \rceil - 1$ **do**
            **for** $q = 0 \rightarrow \lceil \frac{n}{r} \rceil - 1$ **do**
                $x = pr + i + 1$ and $y = qr + j + 1$;
                **if** $x \leq n$ *and* $y \leq n$ **then**
                    **if** *Color* $(c \mod \mathcal{C}) + 1$ *has not been used in any node within* $(r-1)$*-hop neighbor of node* $(x, y)$ **then**
                        Put color $(c \mod \mathcal{C}) + 1$ in node $(x, y)$; $c = c + 1$;
                    **else**
                        Put color $c$ in node $(x, y)$;
                    $\rho = \rho + 1$;
                    **if** $\mathcal{N} \geq \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$ **then**
                        **if** $\rho \geq \mathcal{K} + r'$ **then**
                            **return** $\rho$;
                        **else**
                            **if** $\rho \geq 2\mathcal{K} - (\mathcal{N} - \mathcal{K})(\lceil \frac{n}{r} \rceil^2 - 2)$ **then**
                                **return** $\rho$;

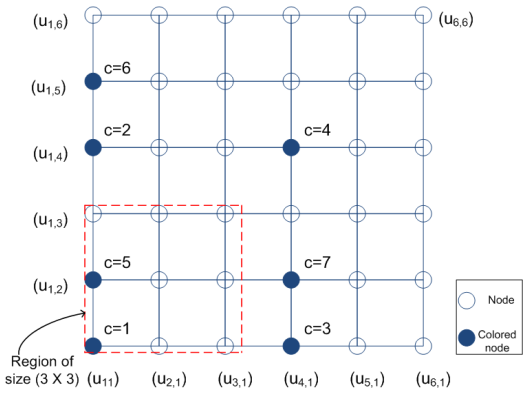



Figure 5: An example of a regular grid network of size $6 \times 6$ and a region fault of size 3 grid. Given $\mathcal{N} = 8$ and $\mathcal{K} = 5$, the example shows how the algorithm colors the node in the network. According to the algorithm colors assigned to node $u_{1,1}$ is 1, $u_{1,4}$ is 2, $u_{4,1}$ is 3, $u_{4,4}$ is 4, $u_{1,2}$ is 5, $u_{1,5}$ is 6 and $u_{4,2}$ is 7

$k'$. Maximum number of non-overlapping regions that can exist simultaneously in a $(n \times n)$ grid is given by $\lceil \frac{n}{r} \rceil^2$. Thus, even if one of these regions fail, the optimal solution should ensure that at least $\mathcal{K}$ colored nodes will be available in rest of the $\lceil \frac{n}{r} \rceil^2 - 1$ regions. Accordingly, the following equation holds: $k' \lceil \frac{n}{r} \rceil^2 \geq \mathcal{K} + k'$ or $k' \geq \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1}$. Optimal solution will find minimum value of $k'$, i.e., $k' = \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$. So if $\mathcal{N} \geq \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$ the total number of nodes that will be colored by optimal solution is equal to $\mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$.

The DDG algorithm colors at most $\lceil \frac{n}{r} \rceil^2$ nodes in the inner *for-loops* (steps 10-31).The algorithm stops after $r'$ iterations of outer *for-loops* (step 7-33). So maximum number of nodes colored in $r'$ iterations is $\rho = r' \lceil \frac{n}{r} \rceil^2$. The algorithm stops when $\rho = \mathcal{K} + r'$. This gives $r' = \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$. So total number of nodes colored ($\rho$) is equal to $\mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$ which is equal to the optimal when $\mathcal{N} \geq \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$.

**CASE II ($\mathcal{K} \leq \mathcal{N} \leq \mathcal{K} + \lceil \frac{\mathcal{K}}{\lceil \frac{n}{r} \rceil^2 - 1} \rceil$):** In this case some of the assigned colors have to be reused. This condition can be divided this case into two subcases

(i) When $\mathcal{N} = \mathcal{K}$, in order to satisfy the color constraint, all $\mathcal{N}$ distinct colors should be available in the residual network. In order to ensure that optimal solution will have a copy of each color assigned to two nodes far enough not to get destroyed by one region fault. In this

case total number of nodes that will be colored is $\rho = 2\mathcal{K}$. The DDG algorithm (in step 14-18) ensures that each color is used at most twice and the nodes with same color do no fall within same region. So total number of nodes colored by the algorithm in this case is $2\mathcal{K}$.

(ii) When $\mathcal{N} - \mathcal{K} > 0$, then each additional available color ($k = \mathcal{N} - \mathcal{K}$) can be treated as a copy of an already assigned color as mentioned in previous case. Since there are a total of $\lceil \frac{n}{r} \rceil^2$ non-overlapping regions, each additional color assigned to a node in one of these regions can be treated as a copy of a color assigned to a node in rest of ($\lceil \frac{n}{r} \rceil^2 - 1$) non-overlapping regions. So total number of nodes that need to be colored in this case is $\rho = 2\mathcal{K} - k(\lceil \frac{n}{r} \rceil^2 - 1) + k$ or $\rho = 2\mathcal{K} - (\mathcal{N} - \mathcal{K})(\lceil \frac{n}{r} \rceil^2 - 2)$. Since the algorithm terminates when $\rho = 2\mathcal{K} - (\mathcal{N} - \mathcal{K})(\lceil \frac{n}{r} \rceil^2 - 2)$ nodes are colored altogether, in this case also, the algorithm produces the optimal solution.

A similar technique can also be used to prove the optimality of the algorithm when $n$ is not a multiple of $r$.

## 2.6 Computational Complexity

In this section, we discuss the computational complexity of the BCDDP. We mention three members of the Hitting Set [25] family of problems.

*Hitting Set Problem* (HS): Given a set of elements $S = \{s_1, \ldots, s_n\}$, another set $\mathcal{S} = \{S_1, \ldots, S_p\}$, where $S_i \subseteq S, 1 \leq i \leq p$, find the smallest subset $S_H \subseteq S$, such that for each set $S_i, 1 \leq i \leq p$, $|S_H \cap S_i| \geq 1$.

*Hitting $\mathcal{K}$-Set Problem*: As described in Section 2.4.

*Budget Constrained Hitting Set Problem* (BCHS): Given a set of elements $S = \{s_1, \ldots, s_n\}$, another set $\mathcal{S} = \{S_1, \ldots, S_p\}$, where $S_i \subseteq S, 1 \leq i \leq p$, and a budget $B$, find a subset

$S' \subseteq S$ of size $B$ that maximizes the size of the subset $\mathcal{S}' \subseteq \mathcal{S}$, such that for each set $S_i \in \mathcal{S}'$, $|S_i \cap S'| \geq 1$.

The BCDDP appears to be very similar to the problems belonging to this family of problems. Since each of these problems are NP-complete, we conjecture that the BCDDP is also NP-complete. Our future work will include the development of a formal proof of hardness for the problem.

## 2.7 Algorithms for the Budget Constrained Data Distribution Problem (BCDDP)

In this section, we first provide an algorithm that is guaranteed to yield an optimal solution for the BCDDP. This is followed by the design and analysis of an approximation algorithm for the same problem.

### 2.7.1 Optimal Solution for the BCDDP in Arbitrary Networks

We can obtain an optimal solution for the BCDDP using the following Integer Linear Program.

For each node $v_i \in V$, define

$$x_i = \begin{cases} 1, & \text{if node } v_i \text{ is assigned a file segment (color)} \\ 0, & \text{otherwise.} \end{cases}$$

For each LCC $C_k \in \mathcal{C}$, define

$$y_k = \begin{cases} 1, & \text{if LCC } C_k \text{ contains at least } \mathcal{K} \text{ file segments} \\ 0, & \text{otherwise.} \end{cases}$$

Then we have the following formulation:

$$\max \sum_{k=1}^{|\mathcal{C}|} y_k$$

$$\mathcal{K} \times y_k \leq \sum_{x_i \in C_k} x_i, \quad \forall k = 1, \ldots, |\mathcal{C}| \tag{2.1}$$

$$\sum_{i=1}^{n} x_i \leq B \tag{2.2}$$

$$y_k \in \{0,1\}, x_i \in \{0,1\};$$

Here, in constraint $(5)$, for each LCC $C_k \in \mathcal{C}$, the corresponding variable $y_k$ is set to be one iff $C_k$ contains at least $\mathcal{K}$ file segments. Constraint $(6)$ ensures that no more than the budget $B$ number of nodes have been selected for storing the encoded file segments. Evidently, the objective function is to maximize the count of the LCCs which are hit at least $\mathcal{K}$ times.

### 2.7.2 Approximation Algorithm for the BCDDP in Arbitrary Networks

We now provide an approximation algorithm for the BCDDP. The input to the algorithm is the following: (i) the layout of graph $G = (V, E)$ on a two-dimensional plane $LG = (Pt, L)$, (ii) the region radius $r$, and (iii) the parameters $B$, $\mathcal{N}$ and $\mathcal{K}$. The output of the algorithm is as follows: A set $V' \subset V$ such that $\mid V' \mid = B$ and if distinct file segments are stored in the nodes of $V'$ then the maximum number of largest connected components of the residual graphs (corresponding to all possible region faults of radius $r$) will have at least $\mathcal{K}$ distinct file segments.

First, using the method described in [23], we compute all the distinct regions $\mathcal{R} = \{R_1, \ldots, R_p\}$ of radius $r$. As noted earlier, there are only $O(n^2)$ and $O(n^4)$ distinct regions for wireless and wired networks respectively [23]. Next, corresponding to every distinct region

$R_i \in \mathcal{R}$, we compute the largest connected component $C_i$ of the residual graph obtained after removal of all the nodes and links in $R_i$. It may be noted that the failure of nodes and links of more than one distinct region may give rise to the same largest connected component. Let $\mathcal{C} = \{C_1, \ldots, C_p\}$ be the set of LCCs corresponding to the distinct region set $\mathcal{R} = \{R_1, \ldots, R_p\}$ of radius $r$. Next, we define a few terms to be used in our algorithm and its analysis.

*Frequency $F(v_i)$ of a node $v_i$*: $F(v_i) =\mid \mathcal{C}' \mid$ where $\mathcal{C}' = \{C_j \in \mathcal{C} \mid v_i \in C_j\}$. Thus, $F(v)$ is the number of $LCC$s that include $v_i$.

*Complete Covering of an LCC $C_i \in \mathcal{C}$ by the node set $V' \subseteq V$*: An LCC $C_i \in \mathcal{C}$ is said to be $completely\ covered$ by $V' \subseteq V$ if $\mid C_i \cap V' \mid \geq \mathcal{K}$.

*Partial Covering of an LCC $C_i \in \mathcal{C}$ by the node set $V' \subseteq V$*: An LCC $C_i \in \mathcal{C}$ is said to be $partially\ covered$ by $V' \subseteq V$ if $\mid C_i \cap V' \mid < \mathcal{K}$.

*Demand of an LCC $C_i$ w.r.t. a node set $V'$ and the coding parameter $\mathcal{K}$:*
Demand($C_i, V', \mathcal{K}$) = $\max(\mathcal{K}$ - $\mid V' \cap C_i \mid$, *0)*, $\forall C_i \in \mathcal{C}$

*Demand Vector $DV_{V',\mathcal{K}}$ w.r.t. a node set $V'$ and the coding parameter $\mathcal{K}$*: a vector of size $\mathcal{K} + 1$, whose $i^{th}$ entry indicates the number of largest connected components (LCCs) that have $demand\ i$ w.r.t. $V'$ and $\mathcal{K}$.

As the goal of the BCDDP is to maximize the number of LCCs that have $\mathcal{K}$ file segments subject to the budget constraint $B$, an intuitive way of solving the BCDDP will be to *greedily* select the $B$ most *frequently* occurring nodes in the set of all LCCs $\mathcal{C} = \{C_1, \ldots, C_p\}$. This is so because by choosing the most frequent $v$'s, *i.e.*, nodes with the highest F($v$) values, the largest number of LCCs $C_i \in \mathcal{C}$ can be "satisfied", at least *partially*, in their requirement that they be "hit" at least $\mathcal{K}$ times. We refer to this process of node selection for file segment storage as a *frequency based approach* (FREQ). Although the frequency based approach is intuitive, it may not always lead to a good solution, as even if an LCC $C_i$ is "hit" $\mathcal{K} - 1$ times, the corresponding region $R_i$ is not made fault-tolerant, because the file cannot be reconstructed

with only $\mathcal{K}-1$ file segments. In some sense, each LCC $C_i \in \mathcal{C}$ has a "demand" to be hit at least $\mathcal{K}$ times so that the benefit of making the corresponding region $R_i$ fault-tolerant is accrued. Selection of a node $v$ in $C_i$ to store a file segment decreases $C_i$'s demand by 1, but unless the demand becomes zero at some point, no benefit is realized. In other words, no credit is received if an LCC is hit fewer than $\mathcal{K}$ times. In a budget constrained scenario, it may so happen that the budget runs out with a large number of LCCs "hit" fewer than $\mathcal{K}$ times and hence no is benefit accrued.

As noted in the previous paragraph, a pure frequency based approach may fail to produce good results as a large number of LCCs may end up storing fewer than $\mathcal{K}$ segments when the budget $B$ runs out. Accordingly, instead of selecting the node with the highest frequency for storing the file segment, we can select the node that hits an LCC with the *least* demand during the node selection process. If at any point of the node selection process, for two LCCs $C_i, C_j \in \mathcal{C}$, $Demand(C_i, V', \mathcal{K}) < Demand(C_j, V', \mathcal{K})$, it implies that $C_i$ has been hit more times by $V'$ than $C_j$. Accordingly, if $\mid V' \mid < B$, selecting a node from $C_i$ is better than selecting a node from $C_j$, as it enhances $C_i$'s chances of being hit $\mathcal{K}$ times (*i.e.*, completely covered) before the budget runs out. We refer to this process of node selection for file segment storage as a *demand based approach* (DMD).

It might appear that if instead of using frequency as the node selection criteria, we use demand as the node selection criteria, it might produce better results. However, this is also not completely true. Consider a scenario with five LCCs, $\mathcal{C} = \{C_1, C_2, C_3, C_4, C_5\}$, where $C_1 = \{v_1, v_2, v_3, v_4\}$, $C_2 = \{v_1, v_2, v_5, v_6\}$, $C_3 = \{v_1, v_2, v_7, v_8\}$, $C_4 = \{v_1, v_2, v_9, v_{10}\}$, $C_5 = \{v_1, v_2, v_{11}, v_{12}\}$, the budget $B = 2$, and the coding parameter $\mathcal{K} = 2$. As $\mathcal{K} = 2$, initially the demand of each LCC $C_i, 1 \leq i \leq 5$, is 2 and as such the demand based algorithm does not make any distinction between the LCCs. The demand based algorithm may choose to store the file segment in node $v_3$, as this selection will partially meet the demand of the LCC

$C_1$. After selection of $v_3$, the demand of $C_1$ changes from 2 to 1, whereas the demand of other LCCs $C_i, 2 \leq i \leq 5$, remains unchanged at 2. Also the available budget decreases from 2 to 1. Since $C_1$'s demand is less than the demand of all other LCCs, the demand based algorithm will choose another node from $C_1$, say $v_1$, for storing the file segment. After this selection, the budget is exhausted and only one LCC (*i.e.*, $C_1$) has two file segments. No other LCC has two file segments which is necessary to reconstruct the file. The optimal solution in this example will be the selection of the nodes $v_1$ and $v_2$, that will meet the demands of all five LCCs, and thus make the file system all region fault-tolerant. From this example, it is evident that taking only demand into consideration for node selection may not lead to a good solution.

| | **N**o. of Connected Components needing $x$ File Segments | | | | |
|---|---|---|---|---|---|
| **I**terations | $x = 0$ | $x = 1$ | $\cdots$ | $x = \mathcal{K} - 1$ | $x = \mathcal{K}$ |
| 0 | 0 | 0 | 0 | 0 | $p$ |
| 1 | 0 | 0 | 0 | $q_1$ | $p - q_1$ |
| 2 | 0 | 0 | $q_3$ | $q_1 + q_2 - q_3$ | $p - q_1 - q_2$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $B$ | $w_0$ | $w_1$ | $\cdots$ | $w_{k-1}$ | $w_k$ |

Table 3: Configuration Table.

From the discussion above, it is clear that both frequency and demand play an important role in the node selection process and neither one alone can produce a good solution. [26] presents an interesting approximation algorithm suitable for addressing reliable network design issues, in particular, the survivable routing problem in WDM mesh networks. However, our problem has a completely different setup as compared to [26]. Accordingly, our approximation algorithm, referred to as a the Hybrid algorithm (HBD), takes both frequency and demand into account in the node selection process. Since no benefit is accrued until the demand of an LCC is met (*i.e.*, $\mathcal{K}$ nodes from the LCC are selected for file segment storage), the hybrid algorithm

assigns more "weight" (or importance) to the demand factor than the frequency factor. HBD is described next in Algorithm 2.

**Algorithm 2:** Hybrid (HBD)

**Input :** 1. The layout of graph $G = (V, E)$ on a two-dimensional plane $LG = (Pt, L)$,
2. Region radius $r$,
3. Parameters $\mathcal{N}$ and $\mathcal{K}$,
4. Budget $B$.

**Output :** A set $V' \subset V$ such that $\mid V' \mid = B$ and if distinct file segments are stored in the nodes of $V'$, then the maximum number of largest connected components of the residual graphs corresponding to all possible distinct region faults of radius $r$ will have at least $\mathcal{K}$ distinct file segments.

```
Compute all the regions R = {R_1, ..., R_p} of radius r in LG using the method
  described in [23];
Find a largest connected component C_i of the residual graph for each region fault
  R_i ∈ R. Let C = {C_1, ..., C_p};
Initialize a Configuration Table CT of size (B+1) × (K+1) with all zeroes;
Initialize first row of CT as {0, 0, ..., p};
V' = φ , iterator = 0;
while | V' | ≠ B do
    foreach v ∈ (V \ V') do
        Initialize a vector DV_{V'∪v,K} of length (K+1) to all zeroes;
        foreach C_i ∈ C do
            m=max ( K− | (V' ∪ v) ∩ C_i | , 0) ;
            Increment DV_{V'∪v,K}[m], 0 ≤ m ≤ K;
        w(V' ∪ v) = Σ_{j=0}^{K} DV_{V'∪v,K}[j] × (p + 1)^{K−j};
    Select a node v ∈ (V \ V') such that w(V' ∪ v) is maximum;
    V' = V' ∪ v;
    Increment iterator;
    Update CT by inserting DV_{V'∪v,K} as the iterator^{th} row of CT;
return V';
```

HBD maintains a *Configuration Table CT*, a $(B+1) \times (\mathcal{K}+1)$ table, to keep track of the "current" demand of every LCC. The $(i+1)^{th}$ row of $CT$ is the Demand Vector $DV_{V',\mathcal{K}}$ w.r.t. $V'$ and $\mathcal{K}$ where $V'$ is the set of nodes selected by HBD during the first $i$ iterations. As such each element $CT_{i,j}$ gives the count of the number of LCCs in $\mathcal{C}$ that have a $demand$ of

$j$ after $i$ nodes have been selected by the algorithm during the first $i$ iterations. An example of a Configuration Table is shown in Table 3. If the LCC set is comprised of $p$ elements, *i.e.*, $\mathcal{C}= \{C_1, \ldots, C_p\}$, then during iteration 0 (*i.e.*, before any node is chosen for file segment storage), the number of LCCs with demand equal to $\mathcal{K}$ is $p$. This is shown in the rightmost entry of row 1 corresponding to Iteration 0. Suppose a node $v_i$ is chosen for storage of a file segment during Iteration 1. If $v_i$ is present in $q_1$ LCCs in the set $\mathcal{C}$, at the end of Iteration 1, the number of LCCs with demand $\mathcal{K}$ will decrease from $\mathcal{K}$ to $\mathcal{K} - q_1$, while the number of LCCs whose demand is $\mathcal{K} - 1$ will increase from 0 to $q_1$. This is shown in row 2 corresponding to Iteration 1 of CT. If the node $v_j$ is selected during Iteration 2, the LCCs that included $v_j$ and were part of the rightmost column (say $q_2$), would move to its immediate left column. Thus, the entry for row 3 corresponding to Iteration 2 in the rightmost column will drop from $p - q_1$ to $p - q_1 - q_2$. The entry in the immediate left column should increase from $q_1$ to $q_1 + q_2$. But, as the node $v_j$ could be a part of the original $q_1$ LCCs (say $q_3$ of them), after selection of $v_j$ the demand of these LCCs will decrease from $\mathcal{K} - 1$ to $\mathcal{K} - 2$. Accordingly, after selection of $v_j$, the number of LCCs with demand $\mathcal{K} - 1$ will change from $q_1$ to $q_1 + q_2 - q_3$ and the number of LCCs with demand $\mathcal{K} - 2$ will change from 0 to $q_3$. This is shown in row 3 corresponding to Iteration 2 of the Configuration Table shown in Table 3.

During each iteration, a node is selected for storing a file segment (*i.e.*, to be included in the set $V'$) and the iteration (node selection) process stops after $B$ iterations, where $B$ is the budget. The decision regarding the selection of the node to be included in the solution set $V'$ is made in the following way. We define *weight of a node set* $V' \subseteq V$ as follows: the weight of the node set $V'$ is $w(V') = \Sigma_{j=0}^{\mathcal{K}} DV_{V',\mathcal{K}}[j] \times (p+1)^{\mathcal{K}-j}$, where $p =\mid \mathcal{C} \mid$. Thus, $w(V')$ is the value of the demand vector $DV_{V',\mathcal{K}}$ when we consider $DV_{V',\mathcal{K}}$ to be a number w.r.t. base $p + 1$. During an iteration, the algorithm selects the node $u$ if augmentation of the node set $V'$ with node $u$ results in the largest increase in the weight of the augmented set $V' \cup u$.

Inside the foreach loop of Steps (7-12) for every node $v \in V \setminus V'$ HBD executes the following computation. First, the *Demand Vector* $DV_{V' \cup v, \mathcal{K}}$ for a set $V' \cup v$ is computed in Steps (7-11). Second, in Step 12 the $weight$ of $DV_{V' \cup v, \mathcal{K}}$ is computed. $w(V' \cup v) > w(V' \cup u)$, if at the index $i$ of the first mismatch from the left hand side for the two vectors $DV_{V' \cup v}$ and $DV_{V' \cup u}$, the value $DV_{V' \cup v}[i]$ is greater than the value $DV_{V' \cup u}[i]$. Since the ultimate goal of the BCDDP is to completely cover as many sets as possible, HBD greedily adds a node $v \in V \setminus V'$ to its final output set $V'$ if the weight of $V' \cup v$ is maximum, among all $v \in V \setminus V'$. This process continues until $\mid V' \mid = B$. These operations are done in Steps (13-16) in HBD. Thus, HBD selects a node $v \in V \setminus V'$ such that maximum number of sets in $\mathcal{C}$ have their demand value reduced and moved towards being completely covered. Hence, HBD takes into account both demand of the sets as well as frequency of the nodes.

#### 2.7.2.1 Performance Analysis of HBD

Given any instance of the BCDDP with budget $B$ and the coding parameter $\mathcal{K}$, let $f_{max}$ denote the frequency of the most frequent node of the graph $G$ in the LCC set $\mathcal{C}$. Also, let $OPT$ be the optimal solution value of this instance and let $HBDS$ be the value of the solution produced by HBD.

**Lemma 2.5.** $OPT \times \mathcal{K} \leq B \times f_{max}$

*Proof.* Consider the left hand side first. By definition of the problem, any LCC in the solution needs to be hit at least $\mathcal{K}$ times. Hence, for any optimal solution, at least $OPT \times \mathcal{K}$ nodes are hit. The right hand side can be interpreted in the following way: we choose exactly $B$ nodes due to the budget constraint and each of them can be hit by at most $f_{max}$ different sets, therefore we can hit no more than $B \times f_{max}$ nodes. Clearly, the LHS should be no larger than the RHS which proves Lemma 2.5. □

**Lemma 2.6.** *The value of the solution produced by HBD, i.e., $HBDS$, is at least $\lfloor \frac{B}{\mathcal{K}} \rfloor$.*

*Proof.* It may be recalled that during the node selection process, due to the weight assignment rules, HBD gives a higher priority to a node that reduces the demand of an LCC with a lower demand value compared to one with a higher demand value. In the worst case scenario, successive rows in the $Configuration\ Table$ have their value in the most significant position incremented by one. But, it still guarantees that after every $\mathcal{K}$ iterations, there will be a new LCC hit $\mathcal{K}$ times. Since there are exactly $B$ iterations, HBD provides a solution with at least $\lfloor \frac{B}{\mathcal{K}} \rfloor$ sets. □

**Theorem 2.7.** *HBD is a $\frac{1}{2f_{max}} - approximation$ algorithm.*

*Proof.* From $Lemma\ 6$, we know $OPT \times \mathcal{K} \leq B \times f_{max}$. Hence, it follows that $OPT \leq \frac{B \times f_{max}}{\mathcal{K}} \leq f_{max} \times (\lfloor \frac{B}{\mathcal{K}} \rfloor + 1)$. From Lemma 2.6, we know $HDBS \geq \lfloor \frac{B}{\mathcal{K}} \rfloor$. Hence, $\frac{HBDS}{OPT} \geq \frac{\lfloor \frac{B}{\mathcal{K}} \rfloor}{f_{max} \times (\lfloor \frac{B}{\mathcal{K}} \rfloor + 1)} = \frac{1}{f_{max}} \times \frac{\lfloor \frac{B}{\mathcal{K}} \rfloor}{\lfloor \frac{B}{\mathcal{K}} \rfloor + 1}$. Since $B \geq \mathcal{K}$ , then $\frac{\lfloor \frac{B}{\mathcal{K}} \rfloor}{\lfloor \frac{B}{\mathcal{K}} \rfloor + 1} \geq \frac{1}{2}$ which proves $\frac{HBDS}{OPT} \geq \frac{1}{2f_{max}}$ □

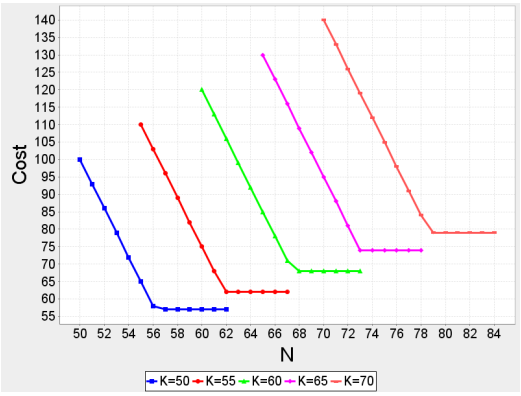
Here we should notice that $\frac{1}{2f_{max}}$ is an asymptotic bound since the tight condition $\frac{B}{\mathcal{K}} = \lfloor \frac{B}{\mathcal{K}} \rfloor + 1$ is never reached. Let $\frac{B}{\mathcal{K}} = \lfloor \frac{B}{\mathcal{K}} \rfloor + \epsilon$ where $0 \leq \epsilon < 1$. Then by the same analysis above, HBD has a bound of $\frac{1}{(1+\epsilon) \times f_{max}}$.

#### 2.7.2.2 Time Complexity Analysis of HBD

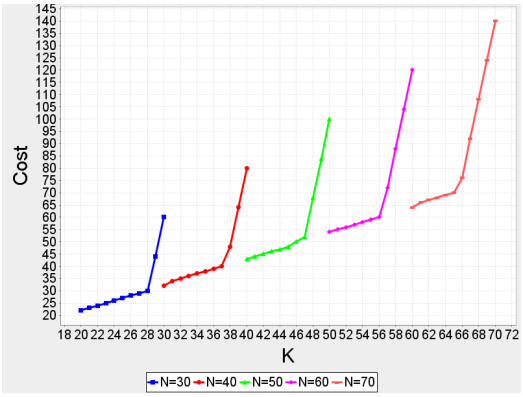
In Steps 1 and 2 of HBD, the largest connected components of all the distinct regions for the layout $LG$ of the graph $G$ can be computed in $O(n^6)$ time using the technique stated in [23]. As $B$ is of $O(n)$, the loop in Steps (6-16) is executed $O(n)$ times. The loop in Steps (7-12) is executed $O(n)$ times. As the number of LCCs is of the order $O(n^4)$, the loop in Steps

(9-11) is also computed the same order of times. The computation in Step 10 takes $O(n)$ time. Accordingly, the overall complexity of the algorithm is $O(n^7)$. Again, the high complexity of the algorithm results because the number of distinct regions that need to be considered is high $O(n^4)$.

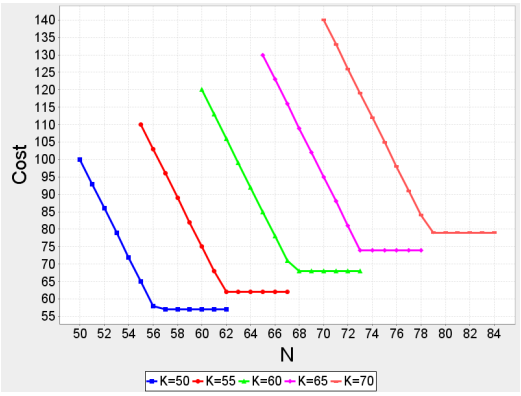
## 2.8 Experimental Results and Discussions for the DDG



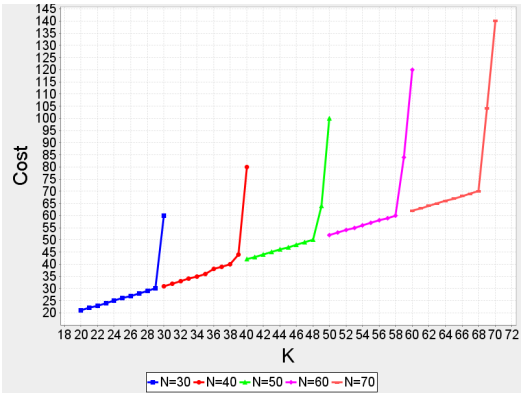

(a) $12 \times 12$ grid, $4 \times 4$ fault region, varying $\mathcal{N}$ over $\mathcal{K}$'s




(b) $20 \times 20$ grid, $5 \times 5$ fault region, varying $\mathcal{K}$ over $\mathcal{N}$'s




(c) $30 \times 30$ grid, $10 \times 10$ fault region, varying $\mathcal{N}$ over $\mathcal{K}$'s




(d) $30 \times 30$ grid, $5 \times 5$ fault region, varying $\mathcal{K}$ over $\mathcal{N}$'s

Figure 6: Experimental results showing impact of coding parameters $\mathcal{N}$ and $\mathcal{K}$ over storage requirement $\sigma$

Using the proposed DDG optimal algorithm we now present our simulation results that demonstrate the impact of the choice of parameters $\mathcal{N}$ and $\mathcal{K}$ on grid networks. The experiments were performed on $n \times n$ grid networks with $n$ of size $12$, $20$, and $30$, with $4$ different fault region sizes for each $n$. For each of the grids, and fault regions chosen, the experiments were categorized and analyzed in two sets, namely – (i) varying parameter $\mathcal{N}$ keeping $\mathcal{K}$ constant, where $\mathcal{K}$ values ranged from $50$ to $70$ in steps of $5$, and (ii) varying parameter $\mathcal{K}$ keeping $\mathcal{N}$ constant, where $\mathcal{N}$ values ranged from $30$ to $70$ in steps of $10$.

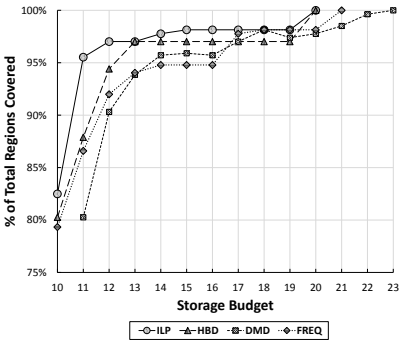
In our experiments we were able to observe that as $\mathcal{N}$ increases from initial value of $\mathcal{K}$, i.e. the ratio between ($\mathcal{N}$, $\mathcal{K}$) increases, storage cost $\sigma$ decreases, whereas when $\mathcal{K}$ increases, i.e. the ratio between ($\mathcal{N}, \mathcal{K}$) decreases, $\sigma$ increases. We were also able to observe that when $\mathcal{N} = \mathcal{K}$, $\sigma$ was $2\mathcal{K}$ confirming CASE II(i) of Theorem 2.4. Also, for a given value of $\mathcal{K}$, our experiments revealed the presence of a threshold value ($\mathcal{T}$) of $\mathcal{N} \leq n$ beyond which $\sigma$ becomes constant for all values of $\mathcal{N} \geq \mathcal{T}$, as shown in Figures 6a and 6c. We were also able to observe that the rate of increase in $\sigma$ as $\mathcal{K}$ approaches $\mathcal{N}$ was much steeper when $\mathcal{K}$ was closer to $\mathcal{N}$, in other words, the rate of increase in storage cost $\sigma$ was not uniform, as shown in Figures 6b and 6d.

## 2.9 Experimental Results and Discussions for the BCDDP

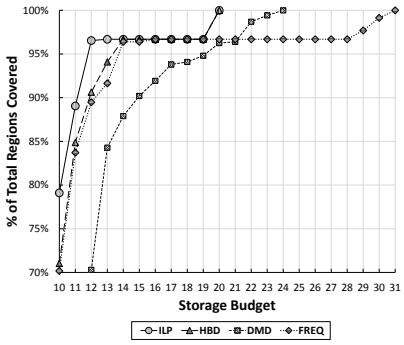
In this section, we present results of our experimentation to demonstrate the efficacy of the proposed approximation algorithm HBD for the BCDDP and we compare its results with the frequency based approach (FREQ) and the demand based approach (DMD) algorithms. Whenever possible, we also show how the solution values of the three algorithms compare against the optimal solution value, obtained by solving an Integer Linear Program formulation of the BCDDP to *completely cover* the maximum number of LCCs with a specified budget.

All the experiments are performed on two real fiber backbone networks of a major network provider [12]: (i) USA network (147 nodes) and (ii) Europe network (46 nodes). The $(x, y)$ coordinates of a node on the network layout is taken to be the latitude and longitude of the corresponding city in the map. A fiber link between two cities in the network map is taken as a straight line between the corresponding nodes in the network layout. All distance units in this section are in latitude and longitude coordinates (*i.e.*, one unit is approximately 60 miles). We consider two parameters that impact the comparison results: $(i)$ the radius of the circular fault region $r$, and $(ii)$ the number of file segments $\mathcal{K}$ required to reconstruct the file. Our experiments have been conducted on a Dell 1955 Linux compute node that is part of a high performance computing cluster. The node is provisioned with 8 cores of 2.66/2.83 GHz processors, 8MB cache, and 16GB of memory. We compute the optimal solution of the BCDDP using CPLEX Optimization Studio 12.5. We denote the optimal solution value as $ILP$.

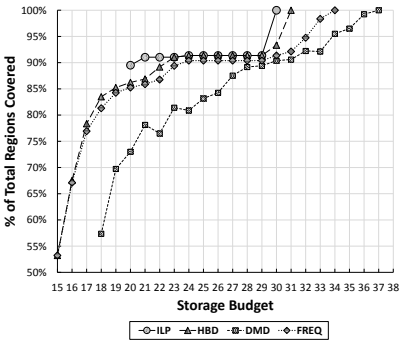
For each of the two networks, our experiments have been carried out on the set of LCCs that have resulted from all distinct fault regions, where the fault radii have been varied from $30$ to $150$ miles in steps of $30$. The LCCs of all the distinct regions for the network layouts of the graph have been computed using the method stated in [23]. In the experiments, $\mathcal{N}$ is taken to be equal to the number of nodes in the graph, and $\mathcal{K}$ is chosen as $5$, $10$ and $15$. For each of the chosen values of $\mathcal{K}$, the budget $B$ for each run of the three algorithms has been varied from $\mathcal{K}$ to the storage cost required by the algorithm to completely cover all LCCs. Since the volume of these results is significantly large, we present in Fig. 7 a selected set of results by varying (i) the budget $B$ and (ii) the percentage of total number of regions covered comparisons between the $HBD$, $DMD$ and $FREQ$ algorithms. Since determining the optimal solution value using the ILP takes on average several hours for execution for each budget allocation, we have been unable to compute the optimal solution for all cases and present results for the optimal solution value in Fig. 7 whenever available. In Figs. 7 (a,b,c), we observe the variation of the number



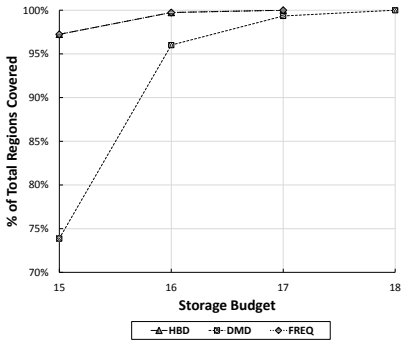

(a) Europe Network, Fault radius $r = 90$ miles, No. of regions $|\mathcal{R}| = 537$, $(\mathcal{N}, \mathcal{K})$=$(46, 10)$



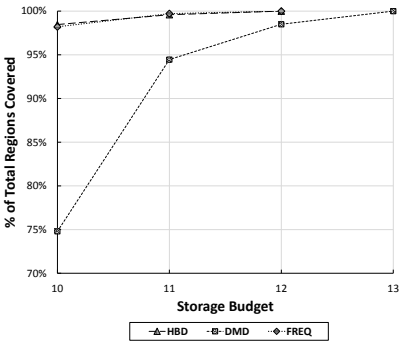

(b) Europe Network, Fault radius $r = 120$ miles, No. of regions $|\mathcal{R}| = 694$, $(\mathcal{N}, \mathcal{K})$=$(46, 10)$



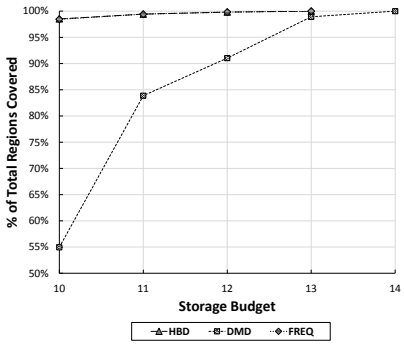

(c) Europe Network, Fault radius $r = 150$ miles, No. of regions $|\mathcal{R}| = 915$, $(\mathcal{N}, \mathcal{K})$=$(46, 15)$




(d) USA Network, Fault radius $r = 60$ miles, No. of regions $|\mathcal{R}| = 2282$, $(\mathcal{N}, \mathcal{K})$=$(147, 15)$




(e) USA Network, Fault radius $r = 90$ miles, No. of regions $|\mathcal{R}| = 3123$, $(\mathcal{N}, \mathcal{K})$=$(147, 10)$




(f) USA Network, Fault radius $r = 150$ miles, No. of regions $|\mathcal{R}| = 4914$, $(\mathcal{N}, \mathcal{K})$=$(147, 10)$

Figure 7: Experimental storage budget vs percentage of total regions covered comparisons between ILP, HBD, DMD, and FREQ solution results.

of resultant LCCs with a change in the fault radius $r$ in the Europe network. In Figs. 7 (d,e,f), we observe the same for the USA network. Because of the resource constraint, we could only perform experiments with small $\mathcal{K}$ values with respect to $\mathcal{N} = 147$ in the USA network.

From the results of our experiments (shown in Fig. 7), we make two observations: (i) Both HBD and FREQ perform significantly better than DMD; (ii) Under certain scenarios HBD performs slightly better than FREQ. In the following paragraphs, we provide explanations for these observations.

Observation (i) can be explained by the fact that the final $V'$ selected by the DMD is heavily dependent on the very first choice of the node $v$ to be included in the set $V'$. All subsequent choices of nodes to be included in $V'$ depend on this choice of $v$. However, the choice of $v$ is random in the sense that at this stage of node selection, the demands of all the LCCs are equal. A bad choice of initial selection may lead the DMD on a bad node selection trajectory, eventually producing a significantly sub-optimal solution.

The explanation for observation (ii) lies with the insight that although both FREQ and HBD start off by selecting the same set of nodes, they eventually diverge in their choice of nodes to be included in $V'$, because unlike FREQ, HBD assigns importance to the $demand$ aspect of the LCCs. In all of the Figs. 7 (a-c), we observe that HBD requires a smaller budget compared to FREQ to achieve the same result, i.e., to achieve $100\%$ coverage of the regions. This implies that FREQ does indeed end up with many *partially covered* LCCs and hence needs a higher budget than HBD to *completely cover* all the LCCs. For example, Fig. 7 (b) shows a particularly bad scenario for FREQ where both HBD and ILP *completely cover* all the regions (694 of them) with a budget of $20$, whereas FREQ is able to cover only 670 of them. The FREQ can provide $100\%$ region coverage only if the budget is increased to 31. In Figs. 7 (d-f), for the USA network, we observe that HBD and FREQ perform identically, in fact they select almost an identical set of nodes. The reason for this behavior is that we have conducted

the experiments for the USA network with values of $\mathcal{K}$ up to fifteen whereas the number of regions is in the order of thousands. For such a small value of $\mathcal{K}$, HBD and FREQ select almost identical sets of nodes because the impact of demand is negligible in this scenario. We were unable to perform experiments in this network with higher values for $\mathcal{K}$ because of the resource constraint - for example, in Fig. 7 (e), the weight function of HBD requires the computation of the first to tenth powers of $p + 1$ where $p = 3123$, the number of regions of the USA network, when the fault radius is 90 miles. Higher values of $\mathcal{K}$ lead to even more intensive computation which is beyond the capabilities of our current computational infrastructure.

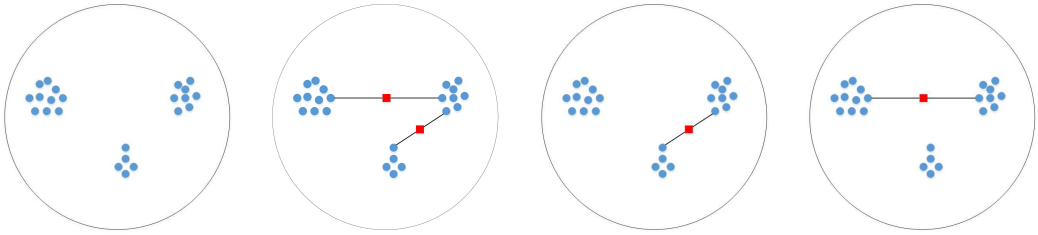
Chapter 3

# BUDGET CONSTRAINED RELAY NODE PLACEMENT PROBLEM FOR MAXIMAL "CONNECTEDNESS"

The relay node placement problem, because of its importance in wireless sensor networks, has been studied fairly extensively in the last few years [27]–[33]. The study of this problem is conducted in a scenario where a number of sensors (nodes) have been placed in a deployment area and often the objective is to place the fewest number of relay nodes in the deployment area such that the resulting network comprising of sensor and relay nodes is *connected*. As the deployment of relay nodes involves *cost*, it may not be possible to acquire and deploy the number of relay nodes necessary to make the entire network connected, particularly when one has to operate under a fixed budget. Although in this scenario, one has to give up the idea of having a network connecting all the sensor nodes, one would still like to have a network with high level of "*connectedness*". In this chapter we introduce the notion of "*connectedness*" for a *disconnected* graph and provide two *metrics* to measure it. The first metric to measure *connectedness* of a disconnected graph is the *number of connected components* of the graph. A *lower number of connected components* in a disconnected graph is an indicator of a *higher degree of connectedness* of the graph. The second metric to measure *connectedness* of a disconnected graph is the *size of the largest connected component* of the graph. A *larger size of the largest connected component* in a disconnected graph is an indicator of a *higher degree of connectedness* of the graph. In this chapter we study the problem whose goal is to design sensor networks with relay nodes to *maximize "connectedness"* subject to a fixed budget constraint. Although resource constrained version of relay node placement problems

have been studied in literature [31]–[33], to the best of our knowledge, the problems investigated in this chapter have not been studied earlier.

The problem scenario studied in this chapter is depicted diagrammatically in Fig 8. By *communication range*, we refer to the upper bound on transmission range. Consider a set of twenty-three sensor nodes (shown as blue circles) deployed as shown in Fig. 8a. Since the mathematical abstraction of the relay node placement problem corresponds to the *Geometric Steiner Tree Problem*, and the terms *Steiner Points* and *terminal points* are used in the abstraction, where the Steiner Points and terminal points correspond to the locations of the relay and sensor nodes respectively, in this chapter we have used the terms "sensor nodes" and "terminal points" interchangeably. In Fig. 8a there are three clusters – the first one with ten terminal points, the second one with eight, while the third with five. The intra-cluster distances are within the communication range, whereas the inter-cluster ones are not. Suppose that the maximum inter cluster distance is less than twice the communication range, and as such only one relay node is sufficient for connecting any two clusters. If we have the option of placing two relay nodes (shown as red squares), then under both metrics of connectedness, the placement of relay nodes as shown in Fig. 8b is an optimal solution. However, if we have a budget of only one relay node, the solution shown in Fig. 8c is an optimal solution under budget constraint according to the first metric of connectedness. This is true as there are exactly two connected components



(a) Deployment of terminal points

(b) Optimal solution with a Budget of 2 relay nodes

(c) Optimal solution for objective 1, but not for objective 2

(d) Optimal solution for both objectives 1 and 2

Figure 8: Figure showing variation in placing relay nodes for different objectives and budget constraints

which is the best that can be achieved with only one relay node. However, in this solution, the largest connected component has thirteen nodes and is not optimal as per the second metric. Fig. 8d shows the optimal placement of the relay node under budget constraint for the second metric, where the largest connected component has eighteen terminal points. It may be noted that this placement also results in an optimal solution under budget constraint according to the first metric. In this dissertation, we have reported our findings using these two metrics. As a future work, we are also considering a unified metric that measures "connectivity" of a disconnected graph by combining the two metrics discussed here.

## 3.1 Problem Formulation

As discussed earlier, the goal of this study is to enhance (or maximize) the "connectedness" of a wireless sensor network with the deployment of a limited number of relay nodes. As a first step in this direction, we formalize the notion of "connectedness" in two different ways, and accordingly, formalize two separate problems. In both problems, we are given: (i) the locations of a set of sensor nodes (terminal points) $P = \{p_1, p_2, \ldots, p_n\}$ in the Euclidean plane, (ii) the communication range $R$ of the sensor and relay nodes, and (iii) a budget $B$ on the number of relay nodes that can be deployed in the sensing field. From the set of points $P$ and communication range $R$, we construct a graph $G = (V, E)$ in the following way. Corresponding to each point $p_i \in P$ we create a node $v_i \in V$ and two nodes $v_i$ and $v_j$ have an edge $e_{i,j} \in E$ if the distance between the points $p_i$ and $p_j$ is at most $R$. It may be noted that the graph $G = (V, E)$ so constructed may be *disconnected* (i.e., it might comprise of a number of *connected components*). The purpose of deploying the relay nodes is to make the *augmented graph*, $G' = (V', E')$, (comprising of sensor and relay nodes) *connected*. Suppose that the $B$ relay nodes are deployed at points $Q = \{q_1, q_2, \ldots, q_B\}$. Corresponding to every point $q_i \in Q$

there is a node $v_i \in V' - V$ and there is an edge between $v_i$ and a node $v_j \in V'$ if the distance between the corresponding points $q_i$ and $p_j$ is at most $R$ ($v_j$ corresponds to $p_j$). With unlimited budget $B$, obviously this goal can be achieved. However, if the budget is smaller than the minimum number of relay nodes necessary to make the graph $G' = (V', E')$ connected, this goal is unachievable. However, in this scenario, one would like to have the graph $G' = (V', E')$ with *as much connectedness as possible*. This gives rise to the "connectedness" maximization problem. The goal of creating the graph $G' = (V', E')$ with *as much connectedness as possible*, can be achieved by (i) deploying the relay nodes in a fashion that *minimizes the number of connected components of* $G' = (V', E')$, or (ii) deploying the relay nodes in a fashion that *maximizes the size of the largest connected components of* $G' = (V', E')$. We refer to (i) as *Budget Constrained Relay node Placement with Minimum Number of Connected Components* (BCRP-MNCC) problem, and (ii) as *Budget Constrained Relay node Placement for Maximizing the Largest Connected Component* (BCRP-MLCC) problem. In BCRP-MNCC, a *smaller number of connected components* is an indicator of a *higher level of connectedness* of the network. While in BCRP-MLCC, a *larger size of the largest connected component* is an indicator of a *higher level of connectedness* of the network. We formally define these two problems as follows:

**Budget Constrained Relay node Placement with Minimum Number of Connected Components** (*BCRP-MNCC*)

Given the locations of $n$ sensor nodes in the Euclidean plane $P = \{p_1, p_2, \ldots, p_n\}$, positive integers $R$, $C$, and a budget $B_1$ on the number of available relay nodes, is it possible to find a set of $Q = \{q_1, q_2, \ldots, q_{B_1}\}$ points in the same plane where relay nodes can be deployed, so that the number of connected components in the graph $G' = (V', E')$ corresponding to the point set $P$ and $Q$ is at most $C$?

**Budget Constrained Relay node Placement with Maximum size of Largest Connected Component** (*BCRP-MLCC*)

Given the locations of $n$ sensor nodes in the Euclidean plane $P = \{p_1, p_2, \ldots, p_n\}$, positive integer $R$, $C$, and a budget $B_2$ on the number of available relay nodes, is it possible to find a set of $Q = \{q_1, q_2, \ldots, q_{B_2}\}$ points in the same plane where relay nodes can be deployed, so that the size of the largest connected component in the graph $G' = (V', E')$ corresponding to the point set $P$ and $Q$ is at least $C$?

The authors in [34] have shown that the *Steiner Tree Problem with Minimum Number of Steiner Points* (STP-MSP) is NP-complete. As STP-MSP problem is a special case of both BCRP-MNCC and BCRP-MLCC problems, and STP-MSP is NP-complete, we can conclude that both BCRP-MNCC and BCRP-MLCC problems are NP-complete. In the following, we elaborate on this point, starting with the formal statement of the STP-MSP problem.

*Steiner Tree Problem with Minimum Number of Steiner Points* (*STP-MSP*): Given a set of $n$ terminals points (location of sensor nodes) $X = \{p_1, p_2, ..., p_n\}$ in the Euclidean plane, and positive integers $R$ and $B_3$, is there a tree $T$ spanning a superset of $X$ such that each edge in the tree has a length of no more than $R$ and the number $C(T)$ of points other than those in $X$, called Steiner points is at most $B_3$? [34]

It may be observed that a special case of the BCRP-MNCC problem where $B_1 = B_3$ and $C = 1$ is equivalent to the STP-MSP problem. Similarly, it may be observed that a special case of the BCRP-MLCC problem where $B_2 = B_3$ and $C = n + B_2$, is equivalent to the STP-MSP problem. Since both BCRP-MNCC and BCRP-MLCC problems are generalization of the STP-MSP problem, we can conclude that both BCRP-MNCC and BCRP-MLCC problems are NP-complete.

## 3.2 Problem Solution

The budget unconstrained version of the relay node placement problem is equivalent to the STP-MSP problem discussed earlier. The authors in [34] have shown that the problem is NP-complete and provided an approximation algorithm with a performance bound of 5. A follow-up paper has since reduced the factor to 3 [35]. The approximation algorithm in [34] follows a *Minimum Spanning Tree* (MST) based approach. Although such an approach provides a constant factor approximation algorithm for the budget unconstrained version of the relay node placement problem, such an approach cannot provide a constant factor approximation algorithm for the budget constrained version of the problem as shown in the example of Fig. 9. In the figure there are three adjacent squares where length of each side is $R' + \epsilon$ and the distance from the circumcenter of the square to a corner point is $R'$.

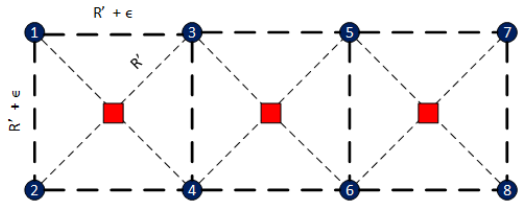



Figure 9: Example to demonstrate that the ratio between the approximate to optimal can be $O(n)$ for any MST based approximation algorithm for BCRP-MNCC problem

In the BCRP-MNCC problem, the goal is to minimize the number of connected components subject to the budget constraint. If we only consider the square with points 1 through 4 and the budget is 1, the optimal number of connected components will be 1 by placing the relay node at the circumcenter of the square. However, in this case the MST based approach will produce 3 connected components as only two of the nodes (1, 2), (1, 3), (2. 4) or (3, 4) can be connected by a single relay node, if the location of the relay node is constrained to be on a line of the MST. Using the same argument, when the locations of the sensor nodes is points 1 through 6 and the budget is 2, the the optimal number of connected components will be 1

whereas the MST based approach will produce 4 connected components. If the locations of the sensor nodes is points 1 through 8 and the budget is 3, the optimal number of connected components will be 1 whereas the MST based approach will produce 5 connected components. In general, such a placement of sensor nodes with $n/2$ nodes on the top row and $n/2$ nodes on the bottom row as shown in Fig. 9 and a budget of $n/2-1$, whereas the optimal placement will produce 1 connected component, the MST based approach will produce $n/2+1$ components. As the the number of sensor nodes $n$ can be arbitrarily large, the ratio between the approximate to the optimal solution of the BCRP-MNCC problem for the MST based approximation algorithm can also grow arbitrarily large.

We next show in subsection 3.2.1 that the computation of the optimal solution of the BCRP-MLCC problem *even when the number of sensor nodes is as few as three is non-trivial*. And in subsection 3.2.2 we provide heuristic solutions for both the BCRP-MNCC and BCRP-MLCC problems where the number of sensor nodes can be arbitrarily large.

### 3.2.1 Optimal Solution for a Special Case of the BCRP-MLCC

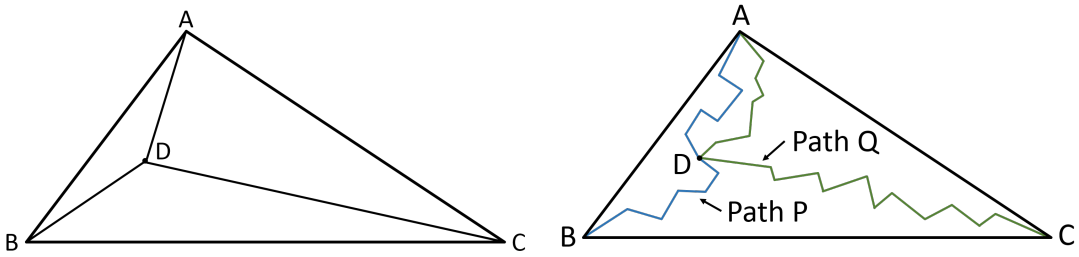
When the number of nodes is 2, i.e., $n = 2$, the BCRP-MLCC problem can be solved trivially. Consider a special case of the BCRP-MLCC problem where $n = 3$, and the distance between each of these nodes is more than the transmission range $R$. W.l.o.g, assume that transmission range for a relay node is 1 unit, i.e. $R = 1$ otherwise we can always divide the length of each side by $R$. Then, for any two nodes $u, v$ on the two-dimensional plane, let $I_{u,v}$ be the interval formed by $u, v$ as end points, and let $|I_{u,v}|$ be the length of the interval, the subsequent observation and lemmas follow:

**Observation 1.** *If we want to make $u$ communicate with $v$ (in isolation w.r.t. other nodes), let the minimum number of relay nodes we need to place be $f(u, v)$, then $f(u, v) = \lceil |I_{u,v}| \rceil - 1$.*

**Lemma 3.1.** *Let $u, v, x, y$ be four nodes on the two-dimensional plane, if $|I_{u,v}| \geq |I_{x,y}|$, then $f(u, v) \geq f(x, y)$.*

**Lemma 3.2.** *If $|I_{u,v}|$ is an integer and $|I_{u,v}| - 1 < |I_{x,y}| \leq |I_{u,v}|$, then $f(u, v) = f(x, y)$.*

Given three nodes $A, B, C$ on the two-dimensional plane, we want to find the minimum number $M$ of relay nodes such that $A, B, C$ can communicate with each other. If $B_2$ is at least $M$, then the optimal solution is 3. Otherwise, the optimal solution is at most 2 and can be computed trivially. Here, we assume $A, B, C$ are not on a straight line, otherwise, the problem can be solved easily by considering two intervals. Therefore, we consider the setting that $A, B, C$ forms a triangle. It may be also noted that if the length of the smallest side of the triangle is at most 1, the problem becomes trivial as well. W.l.o.g, we say that if the side $A, B$ is shorter than 1, then $A, B$ can communicate with each other directly. So we only need to consider link $A, C$ or $B, C$. From Observation 1 the solution will be $\min\{f(A, C), f(B, C)\} = \min\{\lceil |I_{A,C}| \rceil - 1, \lceil |I_{B,C}| \rceil - 1\}$. Hence, we consider scenarios where all side lengths are greater than 1. Evidently, in such a scenario, we need to place at least one relay node and we should place all relay nodes within the triangle area.




(a) Figure depicting the point $D$.

(b) Figure with paths $P$ and $Q$.

Figure 10: Constructions for proof of Claim 1

**Claim 1.** *There exists an optimal solution which contains a relay node $D$, such that all the other relay nodes are located on the intervals $I_{A,D}$ and $I_{B,D}$ and $I_{C,D}$. In other words, the resulting solution looks like a star as shown in Fig. 10a.*

*Proof.* Given any optimal solution, we know the location of all relay nodes. Since $A, B$ can communicate with each other (Fig. 10b), there must be a path $A - B$, path $P = (A = v_1, v_2, ..., v_n = B)$ using relay nodes as intermediate vertices. Similarly, there is an $A - C$ path $Q = (A = u_1, u_2, ..., u_n = C)$. It can be noted that there is no other relay node that is not in $P \cup Q$ as $A, B, C$ is already connected.

We say $D$ is the common node of $P, Q$, in addition, $D$ has the largest index on $P$. Such a $D$ exists since $A = v_1 = u_1$ is a candidate. After obtaining $D$, we divide $P$ into two sub-paths $A - D$ and $D - B$. Since our objective is to minimize the number of relay nodes, both of these sub-paths should be intervals. We consider the same for path $Q$, and the resulting shape looks like a star (in some cases, the resulting shape overlaps two sides of the triangle when $D$ is located at the same location as one of $A$ or $B$ or $C$. □

For any triangle, w.l.o.g, say $(B, C)$ is the longest side with length $L$. Then, it takes at least $\Theta(\lceil L \rceil)$ time to compute the coordinates of all the relay nodes. Next we will present an algorithm that finds the minimum number of required relay nodes in $O(L^2)$ time. The main idea behind the algorithm is to consider the possible options for the optimal location of $D$. We see that once the location of $D$ is fixed, the other relay nodes can be placed greedily at unit distance apart (since $R = 1$) from each other along $I_{A,D}, I_{B,D}$ and $I_{C,D}$ and we can conclude upon the required minimum number of relay nodes for this choice of location of $D$. We categorize the different options of location of $D$ into three major 'Scenarios' which are further divided into different cases. For each setting, we compute the optimal location of $D$ and the total number of relay nodes needed for that choice of $D$. We finally consider the location of $D$ which minimizes the total required number of relay nodes over all categories to obtain the solution for BCRP-MLCC when $n = 3$. The three major scenarios considered are as follows:

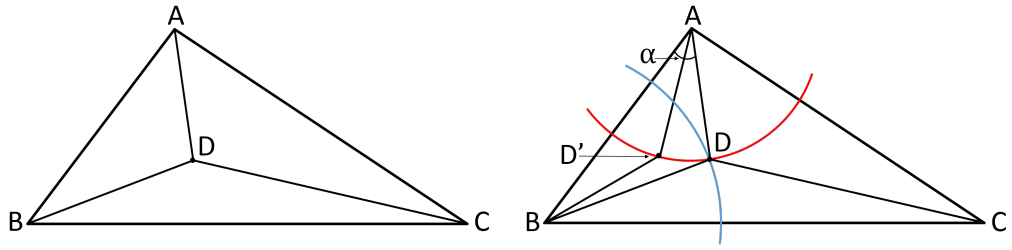
**Scenario 1**: $D$ is located inside the triangle (not on a side).

**Scenario 2**: $D$ is located on side $AC$.

**Scenario 3**: $D$ is located on either side $BC$ or $AB$.

We describe Scenario 1 in details and omit the descriptions and analysis of Scenarios 2 and 3 which follow similarly.

**Scenario 1**: As mentioned earlier, Scenario 1 is when $D$ is located inside the triangle (not on a side) shown in Fig. 11a.




(a) Location of point D (b) Claim 2 Proof Construction

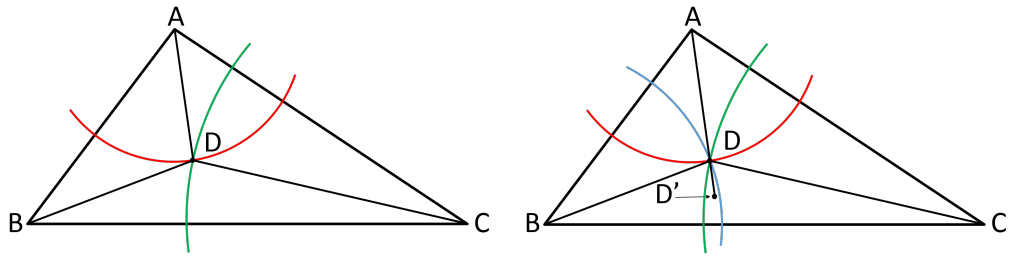
Figure 11: Scenario 1

**Claim 2.** *In scenario 1, there is an optimal solution such that $|I_{C,D}|$ is an integer.*

*Proof.* We pick an optimal solution such that $\alpha = \angle BAD$ is the smallest. Since $D$ is located inside the triangle, $\alpha > 0$. Let $CIR_{P,r}$ be the circle whose centre is $P$ with radius $R$, then $D$ is on the circumference of $CIR_{A,|I_{A,D}|}$ as well as $CIR_{B,|I_{B,D}|}$ as shown in Fig. 11b. Suppose $|I_{C,D}|$ is not an integer, say $|I_{C,D}| = M - \epsilon, M \in \mathbb{N}^+$. Then we can move $D$ along circumference of $CIR_{A,|I_{A,D}|}$ a very small distance, such that $\angle BAD' < \alpha$ and $\angle D'AD \leq \min\{\alpha, \frac{\epsilon}{|I_{A,D}|}\}$. By triangular inequality , $|I_{C,D'}| < |I_{C,D}| + |I_{D,D'}| < |I_{C,D}| + |\overset{\frown}{\text{DD'}}| =\leq M$. According to Lemma 3.2, $f(C, D) = f(C, D')$. Next we consider $I_{A,D'}$. By the construction of $D'$, $|I_{A,D}| = |I_{A,D'}|$ which implies $f(A, D) = f(A, D')$. Finally, we consider $I_{B,D'}$. Since $CIR_{A,|I_{A,D}|}$ intersects $CIR_{B,|I_{B,D}|}$ at $D$, $D'$ must be within $CIR_{B,|I_{B,D}|}$, hence $|I_{B,D'}| < |I_{B,D}|$ and $f(A, D') + f(B, D') + f(C, D') < f(A, D) + f(B, D) + f(C, D)$.

However, based on our choice of $D$ and $\alpha$, this is a contradiction. So, such a $D'$ does not exist and $|I_{C,D}|$ must be an integer. □

Next we show that in Scenario 1 (i) either $|I_{A,D}|$ is an integer, or (ii) one of $\angle ADC$ and $\angle ADB$ is $\frac{\pi}{2}$.




(a) Case (i) (b) Case (ii)

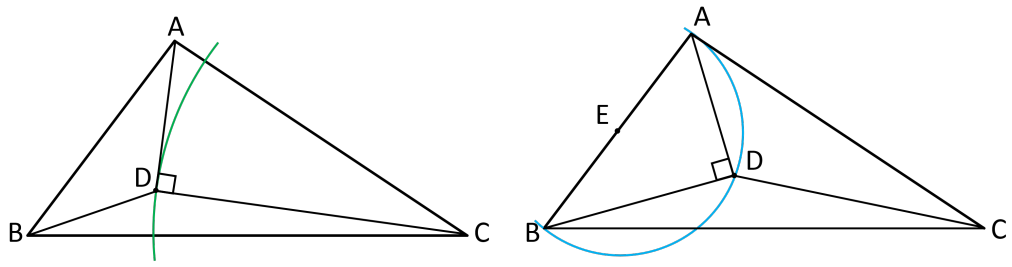
Figure 12: Constructions for Case (i) and Case (ii) under Scenario 1

Case (i): $|I_{A,D}|$ is integral: In this case, as presented in Algorithm 3, we enumerate $|I_{A,D}|$ and $|I_{C,D}|$ (since both are integers), the intersection point (if there are two intersection points, we pick the one inside the triangle) of $CIR_{A,|I_{A,D}|}$ and $CIR_{C,|I_{C,D}|}$ will be the candidate of $D$ (Fig. 12a). Among all candidates, the one that minimizes $f(A,D)+f(B,D)+f(C,D)$ is the final candidate.

**Algorithm 3:** Algorithm to compute scenario 1.(i)

```
for i = 0 to ⌊L⌋ do
    for j = 0 to ⌊L⌋ do
        Compute intersection point, say D, of CIR_{A,i} and CIR_{C,j} if two circle
          intersects;
        if the intersection point is inside triangle then
            Compute f(A,D) + f(B,D) + f(C,D) using f(A,D) = ⌈|I_{A,D}|⌉ − 1
              etc;
Choose D that minimize f(A,D) + f(B,D) + f(C,D), call it D_1;
return;
```

Case (ii): $|I_{A,D}|$ is not integral: By the choice of $D$, $|I_{A,D}|$ cannot be extended. There could be only two reasons for this: either $\angle ADC = \frac{\pi}{2}$, i.e., $AD$ is a tangent line of circle $CIR_{C,|I_{C,D}|}$; or $\angle ADB = \frac{\pi}{2}$, i.e., $AD$ is a tangent line of circle $CIR_{B,|I_{B,D}|}$. This gives rise to the following sub-cases:




(a) Sub-Case 1 (b) Sub-Case 2

Figure 13: Constructions for Scenario 1, Case (ii), Sub-Cases I and II

**Sub-Case I:** $\angle ADC = \frac{\pi}{2}$: As presented in Algorithm 4, we can enumerate over integer values of $|I_{C,D}|$. Then we can compute a tangent $AD$ to $CIR_{C,|I_{C,D}|}$ and get coordinates of $D$ (Fig. 13a). Among all different $D$s, choose the one that minimize minimizes $f(A,D) + f(B,D) + f(C,D)$ as final candidate.

**Algorithm 4:** Algorithm to compute scenario 1.(ii).I

```
for i = 0 to ⌊L⌋ do
    Compute tangent line AD to circle CIR_{C,i};
    if the intersection point is inside triangle then
        Compute f(A,D) + f(B,D) + f(C,D);
Choose D that minimize f(A,D) + f(B,D) + f(C,D), call it D_2;
return;
```

**Sub-Case II:** $\angle ADB = \frac{\pi}{2}$: Let $E$ be the mid point of $AB$, then by knowledge of geometry, $D$ lies on the circumference of $CIR_{E,\frac{|AB|}{2}}$. As presented in Algorithm 5, again we enumerate over integral values of $I_{C,D}$ and compute intersection point of $CIR_{E,\frac{|AB|}{2}}$ and $CIR_{C,|I_{C,D}|}$ (Fig. 13b).

**Algorithm 5:** Algorithm to compute scenario 1.(ii).II

**for** $i = 0$ *to* $\lfloor L \rfloor$ **do**
  Compute intersection point of $CIR_{C,i}$ and $CIR_{E,\frac{\lfloor AB \rfloor}{2}}$ where $E$ is mid point of side $AB$;
  **if** *the intersection point is inside triangle* **then**
    Compute $f(A, D) + f(B, D) + f(C, D)$;
Choose $D$ that minimize $f(A, D) + f(B, D) + f(C, D)$, call it $D_3$;
**return**;

### 3.2.2 Heuristic Solution for BCRP-MNCC with Arbitrary Number of Sensor Nodes

Our heuristic solution for the BCRP-MNCC problem is based on a Minimum Spanning Tree (MST) on the terminal points (sensor nodes). First we construct a weighted complete graph where each node represents a terminal point. The weight of an edge $e$ connecting nodes $v_i$ and $v_j$ is equal to the ceil of the Euclidean distance $length(e)$ between the corresponding terminal points $p_i$ and $p_j$ divided by $R$ and less one, where $R$ is the communication range, i.e., $w(e) = \lceil \frac{length(e)}{R} \rceil - 1$. This weight $w(e)$ represents the number of relay nodes that will be needed to enable communication between the sensor nodes at the two ends of this edge. We then compute an MST on this graph. If the length of an edge of the MST is at most $R$, the two sensor nodes connected by this edge do not need any relay node for communication. However, if the length of an edge of the MST is greater than $R$, some relay nodes will be needed for communication between the sensor nodes connected by this edge. We place the relay nodes on the MST edge (i.e., the line connecting the terminal points) and the number of relay nodes needed to enable communication between two sensor nodes will be equal to $w(e)$.

If the budget on the number of available relay nodes is sufficient, i.e., if $\sum_{e \in E(T')} w(e) \leq B_1$, the number of connected component is one and we directly output the solution. Otherwise, we are short of relay nodes and we successively remove some of the edges of $T'$ till such time that the number of required relay nodes becomes less than or

equal to the budget. It may be noted, removal of one edge from the MST increases the number of connected components by exactly one. We follow a greedy approach for edge removal sequence in that at every stage of the removal process, we remove the highest weighted edge, breaking ties arbitrarily (Algorithm 6).

**Algorithm 6:** Heuristic for BCRP-MNCC problem

1 Create a weighted complete graph $G = (V, E)$ from the set of given terminal points $P$. Assign weight of each edge as $w(e) = \lceil \frac{length(e)}{R} \rceil - 1$;
2 Create an MST $T'$ of $G$ ;
3 **while** $\sum_{e \in E(T')} w(e) > B_1$ **do**
4 Remove the edge that has the maximum weight from $T'$; breaking ties arbitrarily;
5 **return** the resulting forest obtained from $T'$ ;

### 3.2.3 Heuristic Solution for BCRP-MLCC with Arbitrary Number of Sensor Nodes

Our heuristic for the BCRP-MLCC is based on the $k$-MST problem, where one is given an undirected graph $G$ with non-negative costs $c(e)$ for the edges $e \in E(G)$ and an integer $k$, and the problem is to find the minimum-cost tree in $G$ that spans at least $k$ vertices. Computation of $k$-MST is a well studied problem. [36] and [37] present $1 + \epsilon$ approximate solutions for the $k$-MST problem. Our heuristic (Algorithm 7) for the BCRP-MLCC creates a graph $G$ using the same techqniue as in the BCRP-MNCC problem. Next, it computes $k$-MST say $T'$ with decreasing value of $k$ starting with $k = n$, where $n$ is the number of terminal nodes. Once a $k$-MST, say $T'$ is computed, if the weight of the tree $T'$ i.e., $\sum_{e \in E(T')} w(e)$ does not exceed the budget, our procedure stops and outputs the nodes of $T'$ as the largest connected component. Otherwise it computes $k$-MST once again with the value of $k$ decremented by one and then checks if the number of relay nodes needed is within the specified budget.

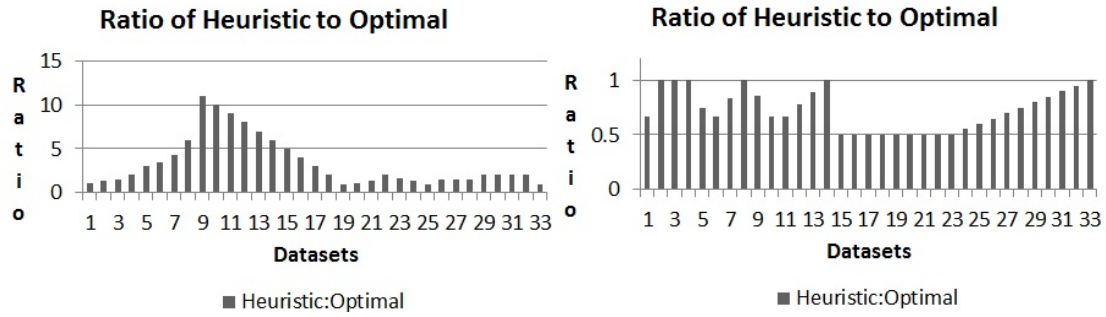
**Algorithm 7:** Heuristic for solving BCRP-MLCC problem

```
Create a weighted complete graph G = (V, E) from the set of given terminal points P.
  Assign weight of each edge as w(e) = ⌈length(e)/R⌉ − 1;
for k = n to 2 do
    Create an approximate k-MST T′ on G ;
    if Σ_{e∈E(T′)} w(e) ≤ B_2 then
        return T′ as the solution of BCRP-MLCC;
return any arbitrary terminal point as solution;
```

### 3.3 Experimental Results

In this section, we present the results of our experimental evaluations of Algorithms 6 and 7. To compute the MST for BCRP-MNCC, we use Prim's algorithm and for $k$-MST for BCRP-MLCC, we use algorithm presented in [38]. In order to evaluate the performance of the heuristics for BCRP-MNCC and BCRP-MLCC problems presented in Algorithms 6 and 7, we need to know both the approximate and the optimal solution for the problem instances. Whereas, approximate (heuristic) solution to the problem instances can be obtained by running Algorithms 6 and 7, optimal solution to the problem instances is not obvious using Integer Linear Programming (which is often used in similar problems scenarios) as the placement of a relay node can be at any point in the deployment area and the number of such points are infinite. To overcome this constraint, we created data sets by placing the sensor nodes at specific locations in the deployment area so that we can compute the optimal solution for a specified budget easily. We manually created 33 datasets, each with 20 sensor nodes and a fixed communication range, varying the (i) the sensor node deployment pattern and the (ii) relay node budget, in a way that we know the optimal solution for these problem instances. The ratio between the heuristic to optimal solution for the BCRP-MNCC and BCRP-MLCC problems are shown in Fig. 14. On the X-axis of Fig. 14 we have data sets 1 through 33 and on the Y-axis have the ratio of the heuristic to the optimal solution for problem instance (i.e., a specific data set). It

may be observed that the ratio between heuristic to optimal was never lower than 0.5 for the BCRP-MLCC problem but for the BCRP-MNCC problem this rato was as large as 11. As we have observed earlier in section 3.2, an MST based solution to the BCRP-MNCC problem can perform poorly, if the sensor nodes have some specific (bad) deployment pattern. The poor performance of the BCRP-MNCC heuristic for some problem instances can be explained by this observation.




(a) Plot of experimental results for BCRP-MNCC problem

(b) Plot of experimental results for BCRP-MLCC problem

Figure 14: Experimental results plotting the ratio of the heuristic to the optimal solutions for different datasets for the BCRP-MNCC and the BCRP-MLCC problems.

Chapter 4

# PROGRESSIVE RECOVERY FROM FAILURE IN MULTI-LAYERED INTERDEPENDENT NETWORK USING A NEW MODEL OF INTERDEPENDENCY

In recent years the research community is becoming increasingly aware of the fact that the critical infrastructures of a nation are heavily interdependent for being fully functional. Let us consider the complex interdependencies that exist between the electric power grid and the communication network. The power grid entities, such as the SCADA systems control power stations and sub-stations. Such SCADA systems receive the critical commands for proper functioning through communication networks. On the other hand, electric power is imperative for communication network entities, such as routers and base stations, to operate.

In order to understand the nuances of the interdependencies between multi-layered networks, the research community has made significant efforts over the past few years [39]–[43]. Although, quite a few models have been proposed to analyze such interdependent networks, most of the models are too simplistic. Thus, unfortunately these models fail to fully capture the complexities pertaining to the interdepedence of power grid and communication networks. In [39], the authors assume that each entity in a network depends on exactly one entity of the other network. However, in a follow up paper [40], the same authors modify this assumption of theirs, simply because in the real world, an entity of a network can in fact depend on multiple entities of the other network.

The generalized model of [40] can account for *disjunctive dependency* of an entity in network $A$ (say $a_i$) on multiple entities in the network $B$ (say, $b_j$ and $b_k$), which implies that $a_i$ may be "alive" (functional) if either $b_i$ or $b_j$ is alive (functional). However, their model still cannot account for *conjunctive dependency* of the form that for $a_i$ to be "alive", *both* $b_j$ *and* $b_k$

must be alive. Furthermore, it is quite likely, that in a real world network the dependency might be even more complex being a combination of both disjunctive and conjunctive components. For e.g., $a_i$ may be alive if (i) $b_j$ *and* $b_k$ *and* $b_l$ are alive, *or* (ii) $b_m$ *and* $b_n$ are alive, *or* (iii) $b_p$ is alive. The graph based interdependency models proposed in [39]–[44], cannot capture such complex interdependency. In order to overcome these shortcomings of the models in the existing literature, we have recently proposed the *Implicative Interdependency Model* (IIM) [45] which uses Boolean logic to capture such complexities.

It may be noted that as entities of network $A$ are dependent on entities of network $B$, which in turn depend on entities of network $A$, the failure of a small number of type $A$ or $B$ entities can trigger a cascade of failures in multi-layered networks resulting in a failure of a large number of entities. Suppose that $V(A) = \{a_1, \ldots, a_n\}$ is the set of entities of network $A$ and $V(B) = \{b_1, \ldots, b_m\}$ is that of network $B$. Further, $A_f^O \subseteq V(A), B_f^O \subseteq V(B)$ represent the subset of $A$ and $B$ type entities respectively whose failure *originally*, results in the failure of $A_f^c \cup B_f^c$ through the cascading failure process. In this case, the set $A_f^O \cup B_f^O$, must be repaired to take the system back from its degraded state to its pre-failure state. Suppose that $A_f^O = \{a_1, \ldots, a_s\}$ and $B_f^O = \{b_1, \ldots, b_t\}$. Every time an element of $A_f^O$ or $B_f^O$ is repaired, the system moves towards its pre-failure state. However, improvement of *system utility* (formally defined in Section 4.2) after repair of an element $a_i \in A_f^O$ (say), may be quite different from that after repair of another element $a_j \in A_f^O$. Accordingly, the *sequence* in which the elements of $A_f^O$ and $B_f^O$ are repaired have significant impact on *s*ystem utility during the recovery process. The goal of the Progressive Recovery Problem is to find the *repair sequence* of the elements of $A_f^O \cup B_f^O$, so that the *s*ystem utility is *maximized* over the entire recovery process. The problem is described in detail in Section 4.2.

We discuss the IIM model in details in Section 4.1. Utilizing the IIM model, we study the progressive recovery problem in an interdependent multi-layered networked system. In

Section 4.2, we formally define the Progressive Recovery Problem. In Section 4.3, we show that this problem can be solved in polynomial time for some special case, whereas for some others, the problem is NP-complete. We provide two approximation algorithms for two special cases of the problem with a performance bound of 2 and 4 respectively. For the most general version of the problem, we provide an optimal solution utilizing Integer Linear Programming and as well a heuristic. Finally, in Section 4.4, we evaluate the efficacy of our heuristic using both synthetic data and real power grid and communication network data collected from Phoenix metropolitan area. The experiments show that our heuristics almost always produce near optimal solution.

## 4.1 Implicative Interdependency Model (IIM)

As mentioned earlier, the *Implicative Interdependency Model (IIM)* [45] was proposed to overcome the limitations of the earlier models [39], [40]. If the network $A$ entity $a_i$ is operational ("alive") if (i) the network $B$ entities $b_j, b_k, b_l$ are operational, or (ii) $b_m, b_n$ are operational, or (iii) $b_p$ is operational, we express this in terms of *live implications* of the form $a_i \leftarrow b_j b_k b_l + b_m b_n + b_p$. Similarly, we can express the live implication for a B type entity $b_r$. We refer to the live implications of the form $a_i \leftarrow b_j b_k b_l + b_m b_n + b_p$ also as First Order Implicative Dependency Relations (IDRs), because these relations express direct dependency of the A type entities on B type entities and vice-versa. It may be noted however that as A type entities are dependent on B type entities, which in turn depends on A type entities, the failure of some A type entities can trigger the failure of other A type entities, though indirectly, through some B type entities. Such an interdependency creates a cascade of failures in multi-layered networks when only a few entities of either A type or B type (or a combination) fail. It may be observed that the IIM model is essentially a Boolean [46]. However, to the best of

our knowledge, such modeling has not been previously used in analyzing progressive recovery techniques in interdependent infrastructure networks.

The IDRs can be formed either through a power-flow analysis of the multi-layer network (similar to the ones carried out by the engineers at FERC [47], and also by the researchers at Columbia University [48] for the power grid), or by consultation with the engineers of the local utility and Internet service providers. It may be noted that it is possible that $A$ type entities may depend on $A$ type entities themselves, similarly, $B$ type entities may depend on $B$ type entities too. The IIM model can deal with such a scenario by not distinguishing between $A$ and $B$ type entities and treating them as a third type entity $C$. Moreover, the concept can easily be generalized to deal with networks with more than two layers.

## 4.2 Progressive Recovery Problem

Let $A_f^O \subseteq V(A), B_f^O \subseteq V(B)$ represent the subset of $A$ and $B$ type entities respectively whose failure initiates a cascade of failures and let $A_f^c \cup B_f^c$ represent the entities that failed due to the *cascading process*. So, the set $A_f^O \cup B_f^O$, must be repaired to take the system from its degraded state back to its *normal functioning state* where all entities should be functional (alive). We call such a set $A_f^O \cup B_f^O$ as the set of *original failures* and for notational simplicity denote it by $D_O$. W.l.o.g, we assume that no IDR has an entity $d_i \in D_O$ on the LHS. Also, the entire set of *failed* entities i.e., $(A_f^O \cup B_f^O) \cup (A_f^c \cup B_f^c)$ is denoted by $D_f$. Suppose that $D_O = \{d_1, d_2, \ldots, d_p\}$, where $d_i \in A_f^O \cup B_f^O$. Obviously, the $p$ entities of the set $D_O$ must be repaired to take the system back to its *normal functioning state*. Suppose that only one failed entity $d_i \in D_O$ can be repaired in one unit of time. Since the real world utilities of the failed entities of $D_f$ may be different, the sequence in which the entities in $D_O$ are repaired

Table 4: IDR's for a Power Communication Network

| **P**ower Net. | **C**omm Net. |
|---|---|
| $a_1 \leftarrow \phi$ | $b_1 \leftarrow a_1 a_2$ |
| $a_2 \leftarrow \phi$ | $b_2 \leftarrow a_1 + a_2$ |
| $\ldots$ | $b_3 \leftarrow a_1$ |

Table 5: $SUOT[T]$ for repair sequence $(a_2, a_1)$

| **T**imestep (t) | 0 | 1 | 2 |
|---|---|---|---|
| $SUIT(t)$ | 0 | 40 | 110 |
| $SUOT[T]$ | 0 | 40 | 150 |

Table 6: $SUOT[T]$ for repair sequence $(a_1, a_2)$

| **T**imestep (t) | 0 | 1 | 2 |
|---|---|---|---|
| $SUIT(t)$ | 0 | 80 | 110 |
| $SUOT[T]$ | 0 | 80 | 190 |

becomes important. We illustrate this with the help of an example. Suppose that the IDRs of an interdependent power-communication network are as given in Table 4.

In this two layer network, when $a_1, a_2$ fail, we see that $b_1, b_2, b_3$ also fail. In order to return the system to its *normal operational state* both $a_1$ and $a_2$ must be repaired. However, whether $a_1$ is repaired first and then $a_2$, or the other way around, will have an impact on *s*ystem utility. Suppose that the utility of an entity $a_i$ is denoted by $u(a_i)$ and is defined as the benefit obtained when the entity $a_i$ is operational. Similarly, we define utility $u(b_j)$ for entity $b_j$. Also, let $x_{a_i}(t)$ be the indicator variable for entity $a_i$ such that $x_{a_i}(t) = 1$ if the entity $a_i$ is operational at time $t$ and 0 otherwise. Indicator variable $x_{b_j}(t)$ is defined similarly for entity $b_j$. We define *System Utility at Instance of Time* $t$, denoted by $SUIT(t)$ as: $SUIT(t) = \sum_{a_i \in V(A)} u(a_i) x_{a_i}(t) + \sum_{b_j \in V(B)} u(b_j) x_{b_j}(t)$, and *System Utility Over Time interval* $0$ *to* $T$ as $SUOT[T]$ as: $SUOT[T] = \sum_{t=0}^{T} SUIT(t)$.

In this example, $D_O = \{a_1, a_2\}$ and $D_f = \{a_1, a_2, b_1, b_2, b_3\}$. Let the utilities of the entities be as follows: $u(a_1) = 10, u(a_2) = 10, u(b_1) = 20, u(b_2) = 30, u(b_3) = 40$. In our analysis, we assume that if an entity $d_i \in D_O$ is fixed at timestep $t$, then all the entities fixed due to the cascade initiated by fixing of $d_i$ are also fixed at timestep $t$, i.e., we ignore the cascade propagation time. If the repair sequence is $a_2$ followed by $a_1$, then $a_2$ and $b_2$ are operational at $t = 1$, and all of $a_1, a_2, b_1, b_2, b_3$ are operational at $t = 2$. If on the other hand, the repair sequence is $a_1$ followed by $a_2$, we have that $a_1, b_2, b_3$ are operational at $t = 1$ and all of $a_1, a_2, b_1, b_2, b_3$ are operational at $t = 2$. The $SUIT(t)$ and $SUOT[T]$ values at different time steps, corresponding to the two different repair sequences are shown in Tables 5 and 6.

From this example, it is clear that the sequence in which the failed entities are repaired has an impact on the *s*ystem utility over time $SUOT[T]$. The *s*ystem utility over time, $SUOT[T]$, for the second sequence $(a_1, a_2)$ is $190$, whereas the $SUOT[T]$ for the first sequence $(a_2, a_1)$ is $150$. Clearly, the second sequence is preferable over the first. The goal of the progressive recovery problem is to identify the repair sequence in such that the *s*ystem utility over time $SUOT[T]$ is maximized.

**Algorithm 8:** Polynomial Algorithm for Progressive Recovery Problem in Case 1

**Input :** (i) A set $S$ of IDR's of implications of the form of $x \leftarrow y$, where $x, y \in V(A) \cup V(B)$, (ii) A set of *o*riginal fault entities $D_O \subseteq V(A) \cup V(B)$, (iii) A set of failed entities $D_f \subseteq V(A) \cup V(B)$, (iv) utility of each entity in $D_f$

**Output :** An ordering $\sigma(D_O)$ such that if the entities of $D_O$ are activated in that order, the value of *SUOT[T]* is maximized.

1 We construct a directed graph $G = (V, E)$, where $V = V(A) \cup V(B)$. For each IDR $x \leftarrow y$ in $S$, where $x, y \in V(A) \cup V(B)$, we introduce a directed edge $(y, x) \in E$;

2 For each node $d_i \in D_O$, we construct a transitive closure set $C_{d_i}$ as follows: If there is a path from $d_i$ to some node $x \in V$ in $G$, then we include $x$ in $C_{d_i}$. We call each $d_i$ to be the *seed entity* for the transitive closure set $C_{d_i}$. This physically means that if $d_i$ fails, all elements in $C_{d_i}$ fail;

3 Sort the *transitive closure sets* ${C_{d_i}}'s$, where the ranks of the closure sets are determined by the sum of the utilities of the failed entities belonging to each closure set. The sets with a larger sum of utilities of failed entities are ranked higher than the sets with a smaller sum. Return the seed entities of the sorted transitive closure sets as the required ordering of the entities of $D_O$;

4 **return**;

## 4.3 Computational Complexity and Solutions

### 4.3.1 Case 1: Problem Instance with One Minterm of Size One

In this case, an IDR in general has the following form: $x_i \leftarrow y_j$ where $x_i$ and $y_j$ belong to networks $A$ ($B$) and $B$ ($A$) respectively. For e.g., in the IDR $a_k \leftarrow b_l$ belonging to Case

1, $x_i = a_k$, $y_j = b_l$. It may be noted that a conjunctive implication of the form $a_i \leftarrow b_j b_k$ can also be written as two separate implications $a_i \leftarrow b_j$ and $a_i \leftarrow b_k$. However, such cases are considered in Case 3 and is excluded from consideration in Case 1. The exclusion of such implications implies that the entities that appear on the LHS of a set of IDRs in Case 1 are unique. So, the in-degree is unity for each node $v \in V(G)$, $G$ being the graph created by Algorithm 8. This property enables us to develop a polynomial time algorithm for the solution of the Progressive Recovery Problem for this case. We present the algorithm next.

*Time complexity of Algorithm 8:* Step 1 takes $O(n + m + r)$ time, where $|V(A)| = n, |V(B)| = m, |S| = r$. Step 2 can be executed in at most $O((n + m)^3)$ time. A standard sorting algorithm in step 3 takes $O(|D_O|log|D_O|)$ time. Since, $|D_O| \leq n + m$, hence the overall time complexity is $O((n + m)^3)$.

**Theorem 4.1.** *For each pair of transitive closure sets $C_{d_i}$ and $C_{d_j}$ produced in step 2 of Algorithm 8, $C_{d_i} \cap C_{d_j} = \emptyset$ where $d_i \neq d_j, d_i, d_j \in D_O$.*

*Proof:* Consider, if possible, that there is a pair of transitive closure sets $C_{d_i}$ and $C_{d_j}$ where $C_{d_i} \cap C_{d_j} \neq \emptyset$. If $C_{d_i} \cap C_{d_j} = C_{d_i}$ or $C_{d_i} \cap C_{d_j} = C_{d_j}$, it means that $d_i \in D_O$ or $d_j \in D_O$ appears on the LHS of an IDR - this is a contradiction to our assumption that $D_O$ is the set of *original* failures. So, let $C_{d_i} \cap C_{d_j} \neq C_{d_i}$ and $C_{d_i} \cap C_{d_j} \neq C_{d_j}$. Let $d_k \in C_{d_i} \cap C_{d_j}$. This implies that there is a path from $d_i$ to $d_k$ ($path_1$) as well as there is a path from $d_j$ to $d_k$, ($path_2$). Since, $d_i \neq d_j$, there is some entity, say $d_l$, in the $path_1$ such that $d_l$ also belongs to $path_2$. It may be noted that $d_l$ may be $d_k$, yet $d_l$ can not be $d_i$ or $d_j$ because in the latter cases either $C_{d_i} \cap C_{d_j} = C_{d_i}$ or $C_{d_i} \cap C_{d_j} = C_{d_j}$. W.l.o.g, let us consider that $d_l$ be the first node in $path_1$ such that $d_l$ also belongs to $path_2$. This implies that $d_l$ has in-degree greater than 1. This in turn implies that there are two IDRs in the set of implications $S$ such that $d_l$ appears

in the LHS of both. This is a contradiction because this violates the characteristic of the IDRs in Case 1.

**Theorem 4.2.** *Algorithm 8 gives an optimal solution for the Progressive Recovery Problem in a multi-layer network for Case 1 dependencies.*

*Proof:* We match the solution $\sigma'$ of Algorithm 8 with the optimal ordering $\sigma_{OPT}$. We say that there is a mismatch at position $r$, when comparing $\sigma'$ with $\sigma_{OPT}$, such that $d_i$ is the $r^{th}$ entity in $\sigma'$, while $d_j$ is the $r^{th}$ entity in $\sigma_{OPT}$ and $\sum_{x \in C_{d_i}} u(x) \neq \sum_{y \in C_{d_j}} u(y)$. So, when comparing $\sigma'$ and $\sigma_{OPT}$, if there are no mismatch as defined, we say that the greedy Algorithm 8 does as good as the optimal solution. Otherwise, let $r$ be the first position of mismatch from the left. By theorem 4.1, we know that $C_{d_i} \cap C_{d_j} = \emptyset$ where $d_i \neq d_j, d_i, d_j \in D_O$. Since, the greedy algorithm did not choose $d_j$ and chose $d_i$ instead, it means that $\sum_{x \in C_{d_i}} u(x) > \sum_{y \in C_{d_j}} u(y)$. So, replacement of $d_i$ with $d_j$ reduces the total number of entities fixed at the $r^{th}$ selection of an entity in $D_O$ to be fixed. This means that the *SUOT[T]* value as achieved by greedy will be more than the optimal solution - this is a contradiction. So, the algorithm in fact returns an optimal solution.

### 4.3.2 Case 2: Problem Instance with Arbitrary number of Minterms of Size One

In this case, the IDRs to be considered are in the general form of $x_j \leftarrow \sum_{i=1}^{k} y_i$, such that $x_j$ belongs to network $A(B)$ and $y_i$ belongs to network $B(A)$. For e.g., $a_p \leftarrow b_q + b_r + b_s$ is an IDR belonging to Case 2.

#### 4.3.2.1 Proof of Hardness

We can show that the *m*in sum set cover (mssc) [49] problem, which is shown to be $NP - hard$, can be reduced to a special case of the Progressive Recovery Problem if all the IDRs are in Case 2. This indicates our Progressive Recovery Problem is also NP-hard if all the IDRs are in Case 2. Following is a brief discussion of the reduction.

**Min sum set cover (mssc )** Viewing the input as a hypergraph $H(V, E)$, a linear ordering is a bijection $f$ from $V$ to $\{1, ..., |V|\}$. For a hyperedge $e$ and linear ordering $f$, defining $f(e)$ as the minimum of $f(v)$ over all $v \in e$. The goal is to find a linear ordering that minimizes $\sum_e f(e)$.

For any instance $I$ in $mssc$, say $H(V, E)$ is the input hypergraph. For each node $v \in V$, we create an entity $b_v$ with $u(b_v) = 0$, and let $D_O = \{b_v | v \in V\}$. For each edge $e \in E$, we create an entity $a_e$ with $u(a_e) = 1$. Also, we create an IDR of the form as $a_e = \sum_{v \in e} b_v$. Clearly, we construct a Progressive Recovery Problem instance in polynomial time with respect to the input size $|V|$ and $|E|$. We denote this instance by $L$. It is easy to check that if we can solve $L$ optimally, we can also obtain optimal solution for $I$, since their objectives are equivalent. Hence, unless $P = NP$, it is impossible to solve Progressive Recovery Problem in polynomial time even if all IDRs belong to Case 2, which means it is $NP - hard$.

#### 4.3.2.2 Optimal Solution using Integer Linear Programming

Here we provide an ILP formulation for the Progressive Recovery Problem. Let $state^t_{a_i}$ (similarly $state^t_{b_j}$) be the indicator variable capturing the state of entity $a_i$ of network A (similarly $b_j$ belonging to network $B$) at timestep $t$. Let $state^t_{a_i} = 0$ if entity $a_i$ is dead at timestep $t$, and $state^t_{a_i} = 1$ if entity $a_i$ is indeed alive at timestep $t$,$1 \leq t \leq |D_O|$. The state variables

for entities $b_j$ of network $B$ are defined likewise. Let the indicator variable $u_t, 1 \leq t \leq |D_O|$ give the value of $SUIT(t)$ (defined in section 3). The objective of the ILP can be written as **maximize** $\sum_{t=1}^{|D_O|} u_t$ where $\sum_{t=1}^{|D_O|} u_t$ gives the value of $SUOT[|D_O|]$ ($SUOT[T]$ is defined in section 3). It may be recalled that the objective of the Progressive Recovery Problem is to find the optimal ordering in which the entities in $D_O$ should be activated such that $SUOT[T]$ is maximized. The constraints of the ILP are as follow:

**1.** $u_t = \sum_{d_i \in D_f} u(d_i) \times state^t_{d_i}, 1 \leq t \leq |D_O|$: This constraint computes the value of $u_t, 1 \leq t \leq |D_O|$ as the sum of the utilities of all the entities which are alive in timestep $t$. Here, $u(d_i)$ gives the utility value for the entity $d_i, 1 \leq i \leq |D_f|$ and is provided as input to the problem.

Now, for each entity $d_i \in D_O$, let the indicator variable $act^t_{d_i} = 1$ if $d_i$ is activated at timestep $t$ and $act^t_{d_i} = 0$ otherwise, where $1 \leq t \leq |D_O|$. So, we have the following constraints:

**2.** $\sum_{t=1}^{|D_O|} act^t_{d_i} = 1, \forall d_i \in D_O$: This constraint ensures that each entity $d_i \in D_O$ is activated exactly once during the time interval $t = 1$ to $t = |D_O|$.

**3.** $\sum_{d_i \in D_O} act^t_{d_i} = 1, 1 \leq t \leq |D_O|$: This constraint ensures that in each timestep $1 \leq t \leq |D_O|$, exactly one entity $d_i \in D_O$ is activated.

**4.** $state^0_{d_i} = 0 \forall d_i \in D_f$: This constraint ensures that at timestep $t = 0$, all entities in $D_f$ are in dead condition.

**5.** $state^t_{d_i} = state^{t-1}_{d_i} + act^t_{d_i}, \forall d_i \in D_O, 1 \leq t \leq |D_O|$: This constraint ensures that the state of an entity $d_i \in D_O$ at timestep $t$ must be the same as that in timestep $t - 1$ unless $d_i$ is activated at timestep $t$. Also, if an entity $d_i \in D_O$ is alive at timestep $t$, it remains alive in timesteps $t + 1, t + 2, \ldots, |D_O|$.

Also, in general form, for each IDR of Case 2, say $x_j \leftarrow \sum_{i=1}^{k} y_i$, we have the following linear constraints. These two constraints ensure that entity $x_j$ is alive only when at least one of $y_i, 1 \leq i \leq k$ is alive.

**6.a** $state_{x_j}^t \geq state_{y_i}^t, 1 \leq i \leq k, 1 \leq t \leq |D_O|$

**6.b** $state_{x_i}^t \leq \sum_{i=1}^{k} y_i, 1 \leq t \leq |D_O|$

For e.g., if we have an IDR of the form $a_1 \leftarrow b_1 + b_2$, for an instance of the Progressive Recovery Problem having $|D_O| = 2$, then this IDR, leads to the following constraints: $state_{a_1}^1 \geq state_{b_1}^1$,$state_{a_1}^1 \geq state_{b_2}^1, state_{a_1}^2 \geq state_{b_1}^2$,$state_{a_1}^2 \geq state_{b_2}^2, state_{a_1}^1 \leq state_{b_1}^1 + state_{b_2}^1$,$state_{a_1}^2 \leq state_{b_1}^2 + state_{b_2}^2$. Thus, given an instance of the Progressive Recovery Problem, we can compute the optimal sequence in which the entities of $D_O$ should be activated by solving this ILP.

#### 4.3.2.3 Approximation Algorithm for a Special Subcase

In our Progressive Recovery Problem, we can transform the IDRs such that the RHS of each IDR consists of only entities of $D_O$. Considering the subcase of our problem such that utilities of all the entities are equal, the objective of this subcase of our problem is identical to that of the $mssc$ problem [49] for which the authors provide a $4-$approximation algorithm.

### 4.3.3 Case 3: Problem Instance with One Minterm of Arbitrary Size

In case 3, the general form of the IDRs to be considered is given by $x_j \leftarrow \prod_{i=1}^{k} y_i$, such that $x_j$ and $y_i$ are entities belonging to network $A(B)$ and $B(A)$ respectively. For e.g., $a_p \leftarrow b_q \times b_r \times b_s$ is an IDR belonging to Case 3.

#### 4.3.3.1 Proof of Hardness

It is possible to show that the *Minimum Latency Set Cover Problem ($MLSC$)* [50], proven as NP-hard, can be reduced to a special case of the Progressive recovery problem if all the IDRs are belong to Case 3.

**The minimum latency set cover problem ($MLSC$)** The problem is defined as follows: Let $J = \{J_1, J_2, ...., J_m\}$ be a set of jobs to be processed by a factory. Each job $J_i$ has a non-negative weight $w_i$. Let $T = \{t_1, t_2, \ldots, t_n\}$ be a set of tools. Job $j$ is associated with a nonempty subset $S_j \subseteq T$ . In each time unit, a single tool can be installed by the factory . Once the entire tool subset $S_j$ has been installed, job $j$ can be processed instantly. The problem is to determine the order of tool installation in order to minimize the weighted sum of job completion times.

Similar construction scheme from Case 2 can be used. Let $I$ be an instance of $MLSC$ and $J$, $T$ be the corresponding input. For each $t \in T$, we create an entity $b_t$ with $u(b_t) = 0$. For each $j \in J$, we create an entity $a_j$ with $u(a_j) = w(j)$. Also, for each $S_j$, we create an IDR of the form as $a_j = \prod_{t \in S_j} b_t$. By arguments similar to those given in Case 2, we know that even Case 3 alone is $NP - hard$.

#### 4.3.3.2 Optimal Solution using Integer Linear Programming

The Interger Linear Programming formulation for the Progressive Recovery Problem when the IDRs belong to Case 3 is almost identical to that when the IDRs are belong to Case 2. The objective function along with constraints one through five remain unchanged. Only constraint 6 changes to account for the change in the form of IDR from Case 2 to Case 3. An IDR in Case 3, in general form, say, $x_j \leftarrow \prod_{i=1}^{k} y_i$ can be represented by the linear constraints

$k \times state^t_{x_j} \leq \sum_{i=1}^{k} state^t_{y_i}, 1 \leq t \leq |D_O|$. These constraints ensure that the entity $x_j$ can be alive only when all the entities $y_i, 1 \leq i \leq k$ are alive. For e.g., let us again consider that we have an IDR $a_1 \leftarrow b_1 \times b_2$ and the instance of the problem has $|D_O| = 2$, then the linear constraints arising from this IDR are $2 \times state^1_{a_1} \leq state^1_{b_1} + state^1_{b_2}$,$2 \times state^2_{a_1} \leq state^2_{b_1} + state^2_{b_2}$. Solving this ILP gives the optimal solution for this case.

#### 4.3.3.3 Approximation Algorithm for a Special Subcase

If $\forall d_i \in D_O$, $u(d_i)$ are equal, we transform the IDRs such that the RHS of all the IDRs are subsets of $D_O$. Then the objective of our problem is identical to that of the $MLSC$ problem. A $2-$approximation algorithm for the $MLSC$ problem is given in [50].

### 4.3.4 Case 4: Problem Instance with Arbitrary Minterm of Arbitrary Size

In the most general setting, an IDR belongs to Case $4$ and has the general form of $x_j \leftarrow \sum_{m=1}^{l} \prod_{i=1}^{k} y_{mi}$, where, as before, $x_j$ and $y_{mi}$ are entities belonging to network $A(B)$ and $B(A)$ respectively. For e.g., $a_p \leftarrow b_q \times b_r + b_s \times b_t$ is an IDR belonging to Case $4$.

#### 4.3.4.1 Proof of Hardness

Because the IDRs belonging to Case $2$ and $3$ are special cases of the general case i.e., Case $4$ and the Progressive Recovery Problem has been proven to be NP-complete when IDRs belong to Case $2$ and $3$, so evidently the problem remains NP-Complete when the IDRs belong to Case $4$ as well.

#### 4.3.4.2 Optimal Solution using Integer Linear Programming

When an IDR belongs to Case 4, it can be expressed in terms of linear constraints by applying a combination of techniques used to translate IDRs belonging to Cases 2 and 3 as discussed in the previous subsections. For e.g., if we have an IDR such as $a_1 \leftarrow b_1 \times b_2 + b_3 \times b_4$, we can re-write it as $a_1 \leftarrow c_1 + c_2$ (and translate it into constraints as discussed in Case 2), where $c_1 \leftarrow b_1 \times b_2$ and $c_2 \leftarrow b_3 \times b_4$ (these are IDRs belonging to Case 3).

#### 4.3.4.3 Heuristic Solution

**Algorithm 9:** Heuristic Algorithm for Progressive Recovery Problem in Case 4

```
set ans = 0;
for i = 1 to n do
    for each node d_i that is not activated yet in D_O do
        Compute influence_{d_i};
        Compute support_{d_i};
    Choose an entity d_i ∈ D_O with the highest influence value influence_{d_i}. If there is
      a tie, choose the one with the larger support value support_{d_i}. Choose one
      arbitrarily if tie still exists. Let e ∈ D_O denote the entity chosen finally;
    Activate e and allow the cascade to occur. Remove any IDR from the set of IDRs if
      the entity on the LHS is fixed at this time step;
    ans = 2 * ans + influence_e;
    σ = σ + e;
return ans and σ;
```

Our heuristic algorithm as given by Algorithm 9 is a greedy one, i.e., we always want to obtain as much utility as possible in each time step. For an entity $d_i \in D_O$, we define $influence_{d_i}$ as the total gain (utility) obtained when entities get fixed following the cascade initiated by activating entity $d_i$ alone. For instance, let us consider the following IDRs: $a_0 \leftarrow b_1, a_1 \leftarrow b_4 + b_2, a_2 \leftarrow b_1 + b_2 \times b_3, b_4 \leftarrow a_0$,$u(a_0) = u(a_1) = u(a_2) = u(b_4) = 1$,$b_1, b_2, b_3 \in$

$D_O$. Then $influence_{b_1} = 4$ since by activating $b_1$, all of $a_0$, $a_1$, $a_2$ and $b_4$ are fixed after cascading. $influence_{b_2} = 1$, for only $a_1$ is fixed upon activation of $b_2$ and $influence_{b_3} = 0$. Entities with higher influence are preferred during each timestep, however, there could be a tie when multiple entities have the same influence value. In order to distinguish, we introduce another variable $support_{d_i}$. For each $d_i \in D_O$, we define $support_{d_i}$ to be the total number of appearances of $d_i$ on the RHS among all IDRs. For instance, if we have $a_0 = b_1 \times b_2, a_1 = b_1 \times b_3, a_2 = b_1 \times b_4$, it is easy to see that all of $b_i$ has influence 0. However, $support_{b_1} = 3$ since it appears thrice on the RHS and $support_{b_2} = support_{b_3} = support_{b_4} = 1$. In particular, if one IDR has the form of $a_1 = b_1 \times b_2 + b_1 \times b_3$, then $support_{b_1} = 1$ for such an IDR. So, whenever there is a tie, we will choose the entity with larger support value. If a tie further exists, we break the tie arbitrarily. Algorithm 9 gives the pseudocode for the heuristic algorithm. Input consists of a set of entities $D_O$ which must be activated, a failed set of entities $D_f$ with utility function $u()$, and IDRs. W.l.o.g, let $|D_O| = n$, $|D_f| = m$ and $r$ is the total number of minterms in the IDR set. Let $\sigma$ be the activation order obtained from Algorithm 9 and $ans$ be the total *s*ystem utility and we recall that we want to maximize the total *s*ystem utility.

The running time of the algorithm is $O(n^2(m+r))$. We need to consider $n$ time steps. During each time step, every entity in $D_O$ is considered. Given an entity $d_i \in D_O$, it takes $O(m)$ time to compute its support value and $O(r)$ time to compute its influence value. Hence the total running time is $O(n^2(m+r))$.

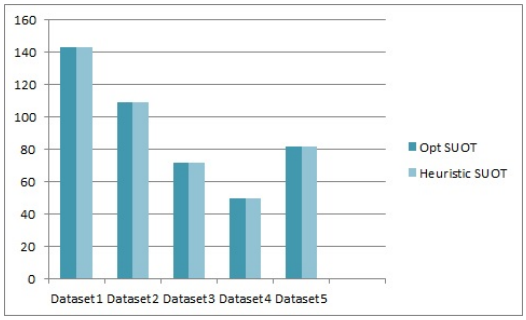
## 4.4 Experimental Result

To study the performance of the heuristic solution for Case 4, we have conducted experiments both on real world data for Phoenix metropolitan area which is the most densely

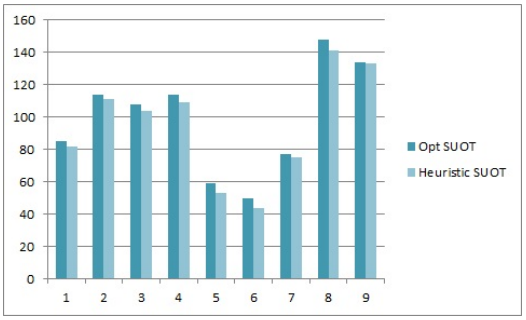
populated area of Arizona, U.S.A, as well as some synthetic data. An overview of the two types of data used in our experiments is as follows:

1. To consider a multi-layer network in a real world setting, we consider the dataset used in our work [45]. We have obtained the data for the power network of Phoenix metropolitan area from Platts (http://www.platts.com/) and that for the communication network from GeoTel (http://www.geo-tel.com/). The power network entities considered are powerplants and transmission lines while the communication network entities considered are fiber-lit buildings, cell towers and fiber links. The dataset consists of $70$ power plants, $470$ transmission lines, $2,690$ cell towers, $7,100$ fiber-lit buildings and $42,723$ fiber links. Due to experimental resource limitation, we have considered five regions of interest in the Phoenix metropolitan area. For each of these five regions, we have constructed a set of IDRs from the power and communication network data using the set of rules described in [45]. For completeness, we describe the set of rules used: *(a)* For each generator to be alive, either the geographically nearest cell tower should be alive or the nearest fiber-lit building and the corresponding fiber link connecting the generator with the fiber-lit building must be alive, *(b)* To be alive, the fiber-lit buildings and the cell towers must have at least one of the two nearest generators and the corresponding connecting transmission lines alive, *(c)* The transmission lines and the fiber links are independent of any other entities.

2. We have also consider twenty datasets of synthetic data. Because of computational resource limitation, for each of these datasets we have considered (1) a random number chosen among $\{2, 3, \ldots, 10\}$ for the size of $D_O$ , (2) a random size for the set $D_f \setminus D_O$ which failed due to cascade, with the sizes varying from ten to twenty, (3) a random number of minterms of random sizes for each IDR. The number of minterms in each IDR is chosen randomly from $\{1, 2, 3\}$. The size of each minterm is randomly chosen from $\{1, 2, \ldots, 8\}$.




(a) Figure comparing optimal and heuristic solutions for the data for the Phoenix metropolitan area




(b) Figure comparing optimal and heuristic solutions for the randomly generated synthetic data

Figure 15: Figure showing experimental comparison of the optimal and heuristic solutions

We have used IBM CPLEX optimizer 12.5 to implement the formulated ILP. We show the results of our experiments both on the real world data as well as the synthetic data in Figure 15. We observe that in the real world data, the heuristic attains optimal solution in each of the five datasets. The reason for such a result is that the IDRs considered are quite simple because of the simple rules [45] as discussed previously- each IDR has at most two minterms and the size of each minterm does not exceed two. In case of the synthetic data, we have considered much more complex IDRs with much bigger sizes for minterms and much bigger failed set $D_f$. However, even in the case of the synthetic data, the heuristic attains near optimal solution in all the cases, with the ratio between the optimal and heuristic solution never exceeding 1.2 in any of the twenty datasets. In Figure 15(b), we compare the optimal and the heuristic solutions for the cases where the latter deviates the most from the optimal solution. It can be thus seen that the heuristic performs quite well in our experimental setup.

Chapter 5

# SPATIO-TEMPORAL SIGNAL RECOVERY FROM POLITICAL TWEETS IN INDONESIA

The sheer popularity of online social media nowadays is reflected by the immense amount of data being fed every second by people from all over the world. It is becoming increasingly evident that analysis of this huge online dataset can provide great insights on the social, political and cultural aspect of the Twitter users and possibly the non-Twitter users as well. In this study, we have developed a tool for recovering spatio-temporal signals from tweets generated in Indonesia. Our interest in analyzing tweets from Indonesia developed in the context of the Minerva[1] project, which was a worldwide project conducted in part at Arizona State University. The goal of this project is to increase the understanding of movements within Muslim communities towards *radicalism* or *counter radicalism*. Based on the *support* and *opposition* of certain *beliefs* and *practices* of an individual (as expressed in her tweet), we can assign a *Radicalization Index* to that individual. In addition, from the self declared *home location* of a Twitter user and the locations of her tweets, we can compute a distribution of *Location Index* for that user. The map of Indonesia is divided up into a set of *regions* and the *Location Index* of a user provides the probability of the user to be in a specific *region* at a specific time. For this analysis a *region* corresponds to a province of Indonesia. Finally, from the *Radicalization Index* and *Location Index* of individuals, *Heat Index* of a *region* , which is a composite measure of the number of radical tweeters of that *region* and their 'degree of radicalism', is computed.

[1]This research was supported in part by US DOD Minerva Research Initiative grant N00014-09-1-0815.

In our model we have a set of tweeters (or users), $U = \{U_1, U_2, \ldots, U_n\}$. Each user $U_i, 1 \leq i \leq n$ creates a set of tweets $T_i = \{T_{i,1}, T_{i,2}, \ldots, T_{i,t_i}\}$. The set of all tweets by all users is denoted by $T = \bigcup_{i=1}^{n} T_i$. The geographic area from where the tweets originate is divided into a set of *regions* $R = \{R_1, R_2, \ldots, R_m\}$. In our study $m$ is equal to thirty four, the number of provinces and special administrative regions of Indonesia. Each user $U_i, 1 \leq i \leq n$ has a *home location* $HL_i, 1 \leq i \leq n$ associated with her, which may or may not be declared. Each tweet $T_{i,k}, 1 \leq i \leq n, 1 \leq k \leq t_i$ has a *geo-location* $GL_{i,k}, 1 \leq i \leq n, 1 \leq k \leq t_i$ associated with it. However, $GL_{i,k}$ for some tweets $T_{i,k}$ may not be known as the user $U_i$ might turn her GPS off. Accordingly, we can divide the set of users in four different classes:

(i) Class 1: user $U_i$ whose *home location* is declared and *geo-location* of at least one tweet is known,

(ii) Class 2: $U_i$ whose *home location* is not declared and *geo-location* of at least one tweet is known,

(iii) Class 3: $U_i$ whose *home location* is declared and *geo-location* of none of the tweets are known, and

(iv) Class 4 : $U_i$ whose *home location* is not declared and *geo-location* of none of the tweets are known.

From the input data set ($U, T, R$ ), we compute, (i) *Location Index*, $L_i$ of each user $U_i, 1 \leq i \leq n$, (ii) *Radicalization Index*, $RD_i$ of each user $U_i, 1 \leq i \leq n$, and finally, combining $L_i$ and $RD_i$, we compute (iii) *Heat Index*, $H_j$ of each *region* $R_j, 1 \leq j \leq m$. It may be noted that whereas $RD_i, 1 \leq i \leq n$ is a scalar value, $L_i$ is a vector of size $m$, $(L_{i,1}, \ldots, L_{i,m})$, where $L_{i,j}$ indicates the probability of user $U_i$ being located in *region* $R_j$ i.e. $L_{i,j}$ indicates the probability of the *Actual home location* of $U_i$ being $R_j$. Finally, the *Heat Index* $H_j$ of region $R_j, 1 \leq j \leq m$ is computed as $H_j = \sum_{i=1}^{n} RD_i \times L_{i,j}, \forall j, 1 \leq j \leq m$. We thus provide a generic technique for generating time-varying political Heat Maps of

a geographical region based on the Twitter data analysis. In this chapter of the dissertation, we have used '*region* ' and 'location' interchangeably to mean an 'Indonesian Province'. It is to be noted that for our calculations, we have considered all Indonesian provinces including special administrative regions such as Yogyakarta and special capital region such as Jakarta and we have ignored Class 4 users.

The rest of the chapter is organized as follows. In Section 5.1, we provide the previous studies related to our work; In Section 5.2, we provide the motivation as well as the distinguishing featurs of this work; Sections 5.3, 5.4, 5.5 present our techniques of computing the Location Indices, the Radicalization Indices and the Heat Indices respectively; In Section 5.6, we discuss our data collection methodology and in Section 5.7, we present our experimental results; Finally, we present the validation of our technique in Section 5.8.

## 5.1 Related Work

Identification of the location of users using Twitter data has been quite a focus of recent research ([51], [52]). [53], [54] combine location information and text from social-network data history to infer user preferences and provide recommendations. However, we do not rely on any 'checking in' information for our computations. 'Geo-coding' (the use of gazetteers) is applicable to our problem since we employ the notion of *regions*. Following [55], we too argue that location estimates are multi-modal probability distributions, rather than particular points or *regions*. However, it may be noted that in contrast to [55], our estimate of the location of the user must be the probability of each Indonesian province as the *Actual home location* of the user under consideration, rather than the probability of the user being located anywhere on the surface of the earth. In this study, we have developed a simple yet effective means

of computing the geo-location of the user as compared to other more complex methodologies such as Topic Detection Techniques [56], [57].

Human mobility is modeled as a stochastic process in [58]. In [59], the authors study the manner in which the movements of human beings are related to time of the day, geography as well as social ties. Similar problems have been studied by [60], [61]. However, in our problem, there is no notion of prediction of location of users involved. Besides, we consider categorical distribution but we apply the concept of mixture of distributions in the lines of [59]. Another line of research which focuses on location estimation by content-analysis of the tweets of a user has been studied by [62], [63]. However, in this current work we rely on the *geo-location* containing tweets of users and their declared *home location* to obtain the location distribution of the users. In [64], the authors analyze tweets generated during the United Kingdom 2010 General Election to infer the political affiliation of a user based on her tweets. We also study a similar problem, however our goal is not to identify the political affiliations of users, rather we compute the 'degree of radicalism' of the user. Besides, unlike them, we apply a very simple yet effective term-frequency analysis of tweets and leverage heavily on our team of domain experts.

The work in [65] which is followed by [66] is very relevant to our technique of *Radicalization Index* assignment to users. These works specifically focus on presenting a framework for combining entity matching techniques for detecting extremist behavior on discussion boards. Identification and analysis of such weak signals of radicalism by the 'lone wolf terrorists' through the use of topic-filtered web harvesting as well as application of natural language processing techniques, thereby fusing aliases for identifying the person form the basis of the works of [65]. Their work is fundamentally different from ours because we deal specifically with the users' publicly available tweets only - this eliminates the availability of the vital background information such as characteristic ('radical internet forum', 'capability internet forum')

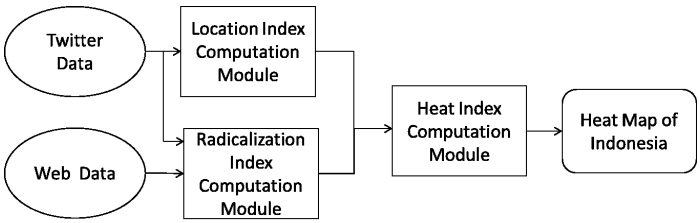



Figure 16: The flow diagram of our Heat Map computation technique. The Web data mentioned here refers to the documents generated by crawling the web pages of radical and counter radical organizations of Indonesia.

annotation of particular discussion boards that is leveraged in [65]. Furthermore, [65] and [66] do not deal with location profiling of users which is one of the two major goals of our work.

## 5.2 Motivation and Distinguishing Features of the Work

The goal of our research is to create a visual description of the spatio-temporal distribution of the radical population of Indonesia by recovering political signals from Twitter data. A flow diagram of our methodology is provided in Figure 16. In [67], the authors retrieve road-kill signals from Twitter data using human beings as sensors. Like [67], we too use human beings as sensors to the extent that we use tweets of Indonesian people to infer radicalism *Heat Indices* of the provinces of Indonesia. However, unlike [67], first we intend to find the distribution of (radical) individuals, so we should not factor in any 'human population bias' i.e variation of densities of people across the different provinces of Indonesia. Second, our problem is much more complex because we not only need to know from which location have the radical tweets come in greater number, but also the 'degree of radicalism' of the tweets - so we need to comprehend the sentiment of the tweets. So, questions of interest to us are-

(Qs1) the 'degree of radicalism' of tweet *tw*

(Qs2) the originating location of tweet *tw*

Thus, *Heat Index* of a region factors in both the count of the radical tweets from the region as well as the 'degree of radicalism' of the tweets. However, there are certain challenges in answering these questions. As for Qs1, a tweet can at most be 140 characters long. This is indeed too little information to ascertain the 'degree of radicalism' of tweets on individual basis. Thus, we go one level up the hierarchy and consider individual users instead of individual tweets. We collect all the tweets from individual users and assign the 'degree of radicalism' to the user based on her tweets. Now, Qs2 would have been easy to answer with respect to individual tweets if all the tweets had geo-co-ordinate information because Twitter API[2] provides *geo-location* information of tweets if the user had chosen to reveal her location at the time of tweeting. However, there are certain problems with this approach - first, the tweets containing *geo-location* information is very scarce (such tweets constitute less than 1% of our dataset). Second, when we consider individual users, it is unjustified to assume that all her tweets containing *geo-location* information will point to a single region, even if all her tweets contained geo-location information. Thus, the best estimate of the location of the user is the probability distribution of the user's location over the Indonesian provinces.

We consider categorical distribution of the users into the thirty four provinces of Indonesia. The motivation behind employing categorical distribution, instead of say the more popular Gaussian distribution over the entire landscape of Indonesia, is that we wish to obtain a political Heat Map of Indonesia with the granularity level of a province. Our technique of *Location Index* computation is discussed in further details in the following section.

Sentiment Analysis using social media data has been attempted by works such as [68] which tries to exploit patterns in online social media communication and also by [69] which uses background lexical information and refining of the same for specific domains by super-

[2]https://dev.twitter.com/docs/streaming-apis and https://dev.twitter.com/docs/platform-objects/tweets have been used

**Algorithm 10:** Counting Algorithm for computation of the *general Computed Home Location* $gCHL$

1 Initialize $gCHL_{a,b} = 0, 1 \leq a \leq m, 1 \leq b \leq m$;
2 For each tweet *tw* in $T$, increment $gCHL_{a,b}$ if *Declared home location* of the author of *tw* and the *geo-location* of *tw* are $R_a$ and $R_b$ respectively;
3 Make each row $gCHL_a$ of $gCHL$ matrix row stochastic, $1 \leq a \leq m$;
4 **return**;

vised learning techniques. However, we have computed *Radicalization Indices* using simpler text regression techniques similar to [70], [71]. Our technique of *Radicalization Index* computation, which is verified to be quite accurate is discussed in further details in Section 5.4.

In summary, individual Twitter users are our chosen level of granularity. We characterize a user not only on the radicalization scale but we also obtain a location distribution of the user over the *regions* of Indonesia. Hence, there is no prediction of the location of the user involved as in [59]. It is to be noted that we consider only the users classified as radical by our *Radicalization Index* computation method.

## 5.3 Location Index Computation

As discussed earlier, each user $U_i, 1 \leq i \leq n$ has a *home location* $HL_i, 1 \leq i \leq n$ associated with her, which may or may not be declared. Each tweet $T_{i,k}, 1 \leq i \leq n, 1 \leq k \leq t_i$ has a *geo-location* $GL_{i,k}, 1 \leq i \leq n, 1 \leq k \leq t_i$ associated with it. However, $GL_{i,k}$ for some tweets $T_{i,k}$ may not be known as the user $U_i$ might turn her GPS off. Even when user $U_i$ has a *Declared Home Location* $DHL_i$, it may not be accurate. User $U_i$ might intentionally or inadvertently misstate her location. Accordingly, we do not accept the $DHL_i$ at its face value as the *Actual home location* of $U_i$. Instead, we compute a matrix, which we term as the *general Computed Home Location* matrix $gCHL$, from the entire dataset barring the timespan (month in our case) for which the Heat Map is being generated. The created matrix $gCHL$ is an $m \times m$

matrix where the entry $gCHL_{a,b}$, $1 \leq a \leq m, 1 \leq b \leq m$, is the *conditional probability* of the *Actual home location* of a user being *region* $R_b$, when her *Declared Home Location* is *region* $R_a$, as learnt from the dataset. The $gCHL$ matrix is computed using the following three steps provided in Algorithm 10. Thus, $gCHL_{a,b}$ is given by:

$$gCHL_{a,b} = \frac{X}{Y}$$

where,

*X* = The number of tweets in $T$ such that the author of the tweet has *Declared Home Location* as $R_a$ and *geo-location* of the tweet is $R_b$

*Y* = The number of tweets in $T$ such that the author of the tweet has *Declared Home Location* as $R_a$

Let the $a^{th}$ row of the $gCHL$ matrix be denoted by $gCHL_a$. Now *Computed Home Location* vector for the user $U_i$ denoted by $CHL_i$ is assigned the value of $gCHL_a$ if $DHL_i$ is *region* $R_a$. It is to be noted that the $gCHL$ matrix is *general* (and not user specific) and is computed using the entire Twitter data set comprising all users.

From those tweets $T_{i,k}, 1 \leq k \leq t_i$ of user $U_i$, that contain the *geo-location* information $GL_{i,k}$, we compute the *Computed Geo Location* vector $CGL_i$ of length m, where $CGL_{i,j}, 1 \leq j \leq m$, is the *probability* of the *Actual home location* of user $U_i$ being *region* $R_j$, as learnt from the tweets of $U_i$. The $CGL_{i,j}$ is computed in the following way:

$$CGL_{i,j} = \frac{A}{B}$$

where,

*A* = The number of tweets in $T_i$ whose *geo-location* is $R_j$ and

*B* = The number of tweets in $T_i$ whose *geo-location* is known

We thus obtain two pieces of information about the *Actual home location* of $U_i$ in the form of two distributions: $CHL_i$ and $CGL_i$, where $CGL_i$ is completely user-specific. How-

ever, $CHL_i$ is partially user-specific - it does depend on $U_i$ because $CHL_i$ is based on her *Declared Home Location*, but it also depends on the general distribution which depends on the entire population mass. It is evident that both $CHL_i$ and $CGL_i$ are categorical distribution over the thirty four Indonesian provinces. Now, a mixture of discrete distributions over any finite number of categories is just another distribution over those categories. In order to combine $CGL_i$ and $CHL_i$ we obtain a convex combination of the two to obtain $L_{i,j}$ in the following way:

$$L_{i,j} = (1 - \omega_i) * CHL_{i,j} + \omega_i * CGL_{i,j} \tag{5.1}$$

Now, the *mixture weights* $\omega_i$ for $U_i$ is learnt from the data itself and is calculated as $\omega_i = |T_i'|/|T_i|$.

$L_{i,j}$ essentially is given by

$$L_{i,j} = |T_i^{''}| * CHL_{i,j} + |T_i^{'}| * CGL_{i,j} \tag{5.2}$$

which gives equation (5.1) when normalized by $|T_i| = |T_i^{'}| + |T_i^{''}|$ i.e the total number of tweets posted by the user $U_i$, where

$T_i$ = set of tweets produced by user $U_i$

$T_i'$ = subset of $T_i$ and represents the set of tweets by $U_i$ that contains *geo-location* information

$T_i''$ = subset of $T_i$ and represents the set of tweets by $U_i$ that do not contain *geo-location* information

The motivation behind this definition of the mixture weight is that for the $T_i^{'}$ tweets which contain *geo-location* information, we consider the user-specific location distribution information inferred from the particular user's *geo-location* containing tweets. However, for the tweets of $T_i^{''}$, we have no location information except for the general information that given a *Declared Home Location* for *any user* $U_v$ in our dataset as $R_a$, $CHL_v$ for $U_v$ is

$gCHL_a$. Thus, if $DHL_i$ of $U_i$ is given to be $R_a$, we consider $CHL_i = gCHL_a$. This simple formulation of $L_{i,j}$ also captures the fact that we rely more on $CGL_i$ than on $CHL_i$ when the number of tweets with *geo-location* information, generated by $U_i$ is high, however if that count is low ( or even absent), instead of discarding $U_i$ 's information, we obtain the location distribution of $U_i$ from her $DHL_i$ . We experimented by using only *geo-location* containing tweets and we saw that the results are far more accurate if we included users of Type 3 - This is intuitively correct because the *geo-location* containing tweets form less than 1% of the entire dataset. We compute $L_{i,j}$ for users belonging to Classes 1-3 (defined previously) using equation (5.1). For the users belonging to Class 3, we obtain $\omega_i$ to be zero, as we do not have any *geo-location* data from the tweets to compute $\omega_i$.

## 5.4 Radicalization Index Computation

We intend to assign a *Radicalization Index* $RD_i$ to $U_i$ based on the *content* of her tweets. We collect tweets from users over a period of time (a month-in our case) and for each user $U_i$ we create a document $D_i$ that contains all the tweets of that user, during that period of time. As there exists a one-to-one correspondence between $U_i$ and $D_i$, by assigning a *Radicalization Index* to $D_i$, we essentially assign $RD_i$ to $U_i$. Classical predictive model Multiple Linear regression [72]–[74] fits our application, since it is a dichotomous classification problem with multiple predictor variables, where the predictor variables are the terms of our "vocabulary". Classical classification methods such as Logistic Regression which has applications in a wide variety of domains can also be used for document classification [75]. Thus, Logistic Regression can also be applied for our problem. However, Linear Regression was selected instead of Logistic Regression because it out-performed the Logistic one through 10-fold cross validation. Linear Regression showed around 98% accuracy, but Logistic Regression showed 83-85% of

Table 7: The table provides the top 5 province or special region names based on their computed Heat Index values (also mentioned alongwith) for October 10 - November 10, November 11 - December 10, December 11- January 10

| Province Name | Heat Index |
|---|---|
| Jakarta | 5.48 |
| East Java | 2.95 |
| West Java | 2.68 |
| Yogyakarta | 1.74 |
| Central Java | 1.68 |

| Province Name | Heat Index |
|---|---|
| Jakarta | 16.16 |
| East Java | 12.33 |
| Yogyakarta | 4.53 |
| Central Java | 3.7 |
| West Java | 3.39 |

| Province Name | Heat Index |
|---|---|
| Jakarta | 4.71 |
| Yogyakarta | 1.82 |
| West Java | 1.25 |
| East Java | 1.20 |
| Central Java | 0.69 |

accuracy. The implementation of our approach proceeds in the following way: First, we identify a set of Indonesian political organizations. Next, social scientists in our Minerva team, who are domain experts for Indonesia, hypothesize a classification to label each organization as radical or counter radical based on these organizations beliefs and practices. Using web crawling tools, we download a large number of documents from the web sites of these organizations. We use the term "vocabulary" to mean the set of all unique terms that appear in all documents from all organizations. All the documents of an organization are assigned the same *Radicalization Index* as that assigned to the organization by the domain experts in our team. This set of documents together with their *Radicalization Indices* form the training dataset for our model. After that we use the model to assign a *Radicalization Index* to the document $D_i$ created from the tweets of user $U_i$. This *Radicalization Index* of document $D_i$ is taken to be $RD_i$ of user $U_i$.

### 5.4.1 Problem Formulation:

We formulate the problem in a general sparse learning framework and solve the following optimization problem (5.3) using the techniques from [76] . This is indeed a sparse learning problem because the vocabulary is very large compared to the number of words used in a document.

$$\min_{x} \frac{1}{2} \|Ax - y\|_2^2 + \frac{\rho}{2} \|x\|_2^2 + \lambda \|x\|_1 \tag{5.3}$$

where $A \in \mathbb{R}^{s \times p}$, $y \in \mathbb{R}^{s \times 1}$, and $x \in \mathbb{R}^{p \times 1}$

In our application, we have

a) $A$ is Document $\times$ Term matrix which is constructed as follows: The *set of terms* $(t_1, \ldots t_p)$ includes all the terms from all the documents by all the organizations, barring the stop words. The size of the vocabulary in this case is $p$. If data is collected by crawling web sites of different organization $(O_1, \ldots, O_q)$ and documents $(d_{i,1}, \ldots, d_{1,r_i})$ are collected from the web site of organization $O_i, 1 \leq i \leq q$, the total number of rows of the matrix $A$ is $s = \Sigma_{i=1}^{q} r_i$ . Thus, $A_{ij} = term\ frequency$ of the $j^{th}$ term in the $i^{th}$ document such that $A_{ij} \geq 0, 1 \leq i \leq s, 1 \leq j \leq p$.

b) $y_i \in \{+1, -1\}$ is the class of each document $D_i, 1 \leq i \leq s$. The *Radicalization Index* of a document is the same the *Radicalization Index* of the organization that created that document. Thus, when an organization is labeled as radical (or counter radical) by the domain experts, all the documents pertaining to that organization is marked as $+1$ (or $-1$). Thus $y_i = +1$(or $-1$) if $D_i, 1 \leq i \leq s$ belongs to an organization marked as radical (or counter radical) by the experts.

c) $x_j$ is the weight for each term $t_j, 1 \leq j \leq p$. This is the parameter estimated by optimizing the objective function (5.3). The $x_j$'s thus form the predictor variables of the model.

Let us further clarify the three terms involved in the convex optimization problem:

a) $\frac{1}{2} \|Ax - y\|_2^2$ - this first term is related to the sum of the squared errors to fit a straight line to a set of data points. The objective function (5.3) thus is the optimization problem of minimizing this sum of squared-errors.

b) $\frac{\rho}{2} \|x\|_2^2$ - this term deals with the ridge regression, which is an extra level of shrinkage. We set $\rho = 0$ as we were mainly driven by sparsity.

c) $\lambda \|x\|_1$ - this term involving the $L1$ norm deals with the sparsity of the solution vector $x$. For different values of $\lambda$ we obtain a solution vector $x$ which represents the weights associated with each term $t_j, 1 \leq j \leq p$ ( the same terms which are considered in the $A$ matrix). Some of these weights are positive, some negative (values can be very close to $0$). The terms with positive (or negative) weights are the radical (or counter radical) words. The top (ones with weights having high magnitude) radical and counter radical words are presented to the experts for validation. We experiment with several $\lambda$ values resulting in $x$ vectors of various sparsity until the list of top radical and counter radical words are approved by the field experts.

We use the Matlab implementation of the SLEP package [77] that utilizes gradient descent approach to solve the optimization problem (5.3). This package can handle matrices of 20M entries within a couple of seconds on a machine with standard configuration. The input to the SLEP package are the values of $A$, $\lambda$, and $y$. The SLEP model outputs the weight vector $x$.

### 5.4.2 Assignment of Radicalization Index:

For each time period (in our case one month), each user $U_i$ will be assigned an $RD_i$ based on their tweets within that period. This is done as follows:

a) As mentioned earlier, from the tweets of each user $U_i$ we form a User Document $D_i$. It is to be noted here that many users choose to tweet quite infrequently, hence even if we collect tweets for one month, a user might have tweeted only once or twice during the entire one month which defeats the purpose of collecting tweets for a month. Hence, we further apply the constraint that we consider only those users who have tweeted at least seven times in a month. The value of this threshold has been arrived at empirically after experimentation with various values of the threshold.

b) With the help of the model that has been fitted using the organization documents, we classify the $D_i$'s. Let each $D_i$ which is a $term\ frequency$ row be denoted by the row vector $t_c$ of count of terms from our "vocabulary".

c) Each user $U_i$ receives a 'score' which we refer to as $RD_i$ given by $RD_i = t_c.x = \sum_{j=1}^{p} t_{c_j} x_j$.

This provides us a time-series of $RD_i$ values for the users. This makes it possible to analyze the transition dynamics of each user. Evidently, a high positive $RD_i$ indicates that $U_i$ is highly radical whereas a high negative $RD_i$ indicates that $U_i$ is highly counter radical.

## 5.5 Heat Index Computation

Once we have obtained the *Location Indices* $L_i, 1 \leq i \leq n$ and *Radicalization Indices* $RD_i, 1 \leq i \leq n$, for all the users $U_i$ , $1 \leq i \leq n$ , the *Heat Index* $H_j$ of region $R_j, 1 \leq j \leq m$ is computed as $H_j = \sum_{i=1}^{n} RD_i \times L_{i,j}, \forall j, 1 \leq j \leq m$. The *Heat Index* $H_j$ for a *region* $R_j$

indicates the degree of prevalence of radical ideologies among the people of $R_j$ by taking into account both the number of radical tweeters living in $R_j$ and also their 'degree of radicalism'.

## 5.6 Data Collection

Since our model requires the computation of both $RD_i$ as well as $L_i$ for each user $U_i$, we followed a two step data collection procedure described as follows:

i) For the purpose collecting the training data set for computing the *Radicalization Index*, we crawled the websites of 36 well-known Indonesian organizations which are classified as radical or counter radical by our field experts. A few of the organizations are mentioned in Table II. We crawled the websites of all these different organizations and collected a total of 78,135 documents which after pre-processing and filtering resulted into 49,250 documents. The reason for this reduction in numbers is that many of the crawled documents did not have any relevant information (for example documents having only advertisements). Each of the documents on a average contained 280 words i.e on an average 2880 characters. All documents pertaining to an organization were labeled as radical or counter radical depending on the outlook professed by the organization itself. These were then used for fitting our *Radicalization Index* computation model.

ii) For our study on recovery of political signals pertaining to trend of radical activities in Indonesia, we chose Twitter as the data collection platform as Indonesia accounts for 19.0% to 20.8% of Twitter's total reach by country (Dec 2010)[3]. No other publicly available portal offers access to opinions posted online by the Indonesian populace on a similar scale as does Twitter. For gathering tweets, we use Twitter's Stream API to access Twitter's global stream

[3]http://www.billhartzer.com/pages/comscore-twitter-latin-america-usage/
http://www.comscoredatamine.com/2011/02/the-netherlands-leads-global-markets-in-twitter-reach/

| Radical Organizations | Counter radical Organizations |
|---|---|
| AbuJibriel | NU |
| PKS | Interfidei |
| Arrahmah | IslamLiberal |
| EraMuslim | PPIM |
| HizbutTahrir | LKIS |

Table 8: Table showing some of the well-known radical and counter radical organizations of Indonesia

| Keyword | Interpretation |
|---|---|
| “penegakan syariah” | enforcement of Sharia |
| “jihad majelis” | jihad assemblies |
| “mati syahid” | martyrdom |
| “ajaran islam” | the teaching of Islam |
| “pendidikan agama di sekolah” | religious education in schools |
| “demokrasi yang” | democracy |

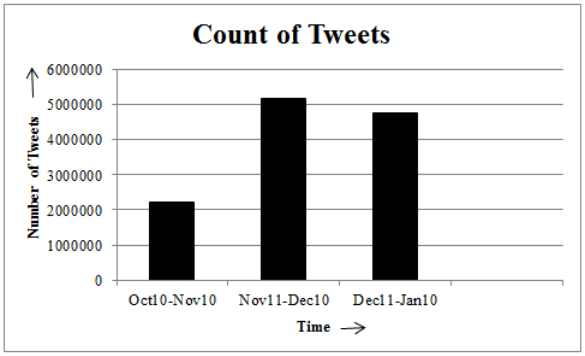
Table 9: Keyword markers used for filtering Twitter Stream API

of publicly available tweet data. Since our goal is to recover “political signals”, we setup a keyword filter on the Stream API to gather tweets that relate to radical and counter radical ideologies. The keywords used for this filtration have been identified by the social scientists in our Minerva project team and are considered to be significant markers of radical and counter radical ideologies in the Indonesian context. A few such markers are listed in Table 9.

We collected tweet data for a three-month interval and gathered a total of 12,152,874 tweets from October 10, 2012 to January 10, 2013 ( Figure 17) that matched the keyword filtration criteria. In this research, we are interested in the probability distribution *Location Index* $L_i$ of user $U_i$ over the thirty four provinces of Indonesia, thus we focus only on users from Indonesia. The keywords used are in Indonesian language and narrows down the tweets we obtained from the Twitter API. Thus, the geo-code in majority of cases indicated a location in Indonesia. However, not all geo-codes are from Indonesia. We ignore those tweets in the current work. Thus, out of these 12 million tweets, 110,063 tweets contained *geo-locations* that mapped

to *regions* within Indonesia. To apply this *reverse geo-coding*, we used the OpenStreetMap API.[4]. A user repository was constructed by including only those users whose *Declared Home Locations* matched with an identifiable Indonesian city or province. We found that many users have put texts such as "Dark side of the moon" or "Here" or "infront of my laptop" as *home location*and hence, there is a need for pre-processing of the text. Also, the users provided location information to varied degrees of granularity ranging from continents to towns. However we are interested in the fixed granularity level of Indonesian provinces and the special regions such as Jakarta and Yogyakarta. Hence we manually created a database of towns and cities of all of the Indonesian provinces. Each of the provinces were annotated with 42 cities/ towns on an average with Papua being the highest which was annotated with 70 cities/towns. Using this database we then assigned a legitimate *Declared Home Location* to as many users as possible. The final user repository consisted of 959,911 unique users.

Figure 17: Figure showing the number of tweets collected over our observation period

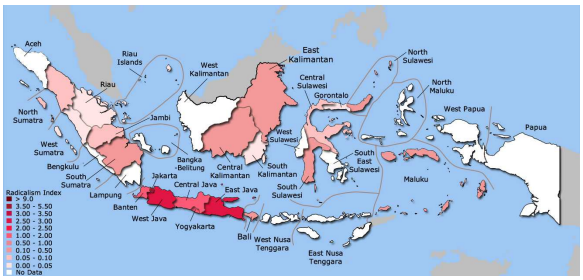



[4]The relevant information about the API could be found at http://wiki.openstreetmap.org/wiki/Nominatim

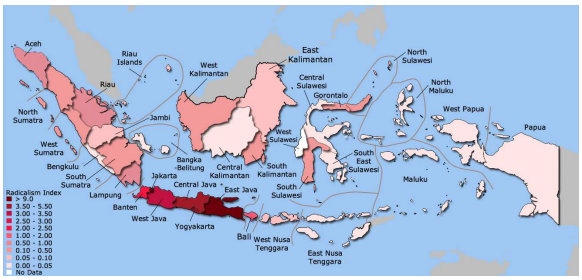
## 5.7 Experimental Results

We created Heat Maps of Indonesia on a monthly basis. We computed the $RD_i$ of each user $U_i$ for each month from October 10 to January 10, as long as $U_i$ tweeted at least 7 times in that month Again, for each user $U_i$ we computed the *Location Index* $L_i$ by considering all her tweets over the period of the month.. For that we computed the *general Computed Home Location* $gCHL$ matrix. The $gCHL$ matrix provides interesting insights on the Indonesian population. We computed the $gCHL$ matrix on all possible doublets among the three months of observation period. i.e for each month for calculating the *Location Indices* $L_i$ of users, we have generated the $gCHL$ matrix using the other two months of data. Thus, in each case, we had training data of two months and test data of one month. We observed that people with *Declared Home Locations* in various different provinces from all around Indonesia such as Bangka Belitung, Banten, Maluku, West Nusa Tenggara, East Nusa Tenggara and Papua have a very strong tendency to have high probability of having *Actual home location* in Jakarta (as observed from our results over three months). This is very intuitive because Jakarta being the Capital Region must have attracted people from different parts of Indonesia for prospective settlement. We further made an observation that people with *Declared Home Location* of East Kalimantan have considerable *geo-location* containing tweets from Central Kalimantan.

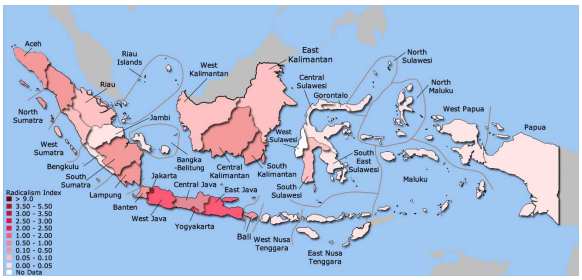
The *Heat Indices* values for the thirty four Indonesian provinces are computed using our approach for three months of our observation period - namely October 10 - November 10, November 11 - December 10, December 11 -January 10. We found a drastic change in the *heat indices* during the interval of November 10 – December 10. But we could not discern any particular event which could have triggered the same. Among all Indonesian provinces the top five provinces and special regions along with their *Heat Index* values are presented in Table I for the three months. Color maps of Indonesia with *Heat Indices* is shown in Figure



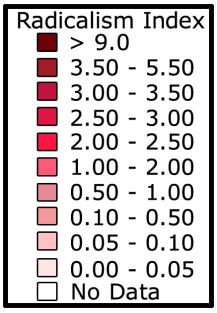

(a) Heat Map for October 10 to November 10




(b) Heat Map for November 11 to December 10




(c) Heat Map for December 11 to January 10




(d) Radicalism Index Scale

Figure 18: Heat Maps of Indonesia

18, where darker colors indicate a higher level of radical tweeting, and lighter colors indicate a lower level of radical tweeting. It may be seen from Figure 18 that the area around Jakarta and the Java provinces are highly active in radical tweet creation. According to our Twitter data analysis, the provinces Jakarta, East Java, Yogyakarta and Central Java, along with West Java are the top provinces that generate a high level of radical activities.

## 5.8 Validation

For the purpose of validation of the *Radicalization Index* (not the 'degree of radicalism'), we computed the *Radicalization Indices* of some well-known counter radical leaders of Indonesia for the months that they had tweeted for more than 7 times which we consider as our threshold. Our classifier gave perfect accuracy. By accuracy of the classification we mean the percentage

of time the leaders who are thus known to be counter radical were classified as counter radical by our classifier. We did not validate the *Location Index* computation technique because of the lack of the ground truth of the *Actual home location* of users. However, our results of Heat Index are validated by the findings of the Indonesia-based Wahid Institute[5]. Wahid Institute promotes a moderate version of Islam through dialogue events, publications, and public advocacies. According to the Wahid Institute's Annual Report of 2012[6], the top four provinces of Indonesia where radical activities are most observable are West Java, Aceh, East Java, and Central Java. It may be noted here, that three out of the four most radical provinces identified by the Wahid Institute, also appear at the very top of our list. Also, our field experts have confirmed Jakarta to be a center of radical activities. It may be mentioned here that field studies[7] in January 2012 by Setara Institute[8], a well-known Indonesian NGO, showed that the strong radicalism of the young muslim population in Yogyakarta and Central Java are making them hot targets to be recruited as Jihadists. In May 2012, there was a mob attack by Indonesian Mujahidin Council in Yogyakarta and in September 2012, there has been arrests of potential terrorists from Yogyakarta[9]. Because, Wahid Institute has mentioned about Indonesian provinces only, it might be expected that Jakarta and Yogyakarta, being special administrative regions, are missing from their list - however, we do not have access to their full report. The high radicalism of the Java provinces are also corroborated by reports of the Setara Institute. The only radically active province that shows up in the Wahid Institute report but does not appear

[5]http://berkleycenter.georgetown.edu/resources/organizations/wahid-institute

[6]Released on December 28, 2012

[7]http://www.setara-institute.org/en/content/study-shows-how-young-radical-indonesian-muslims-become-terrorists

[8]http://www.setara-institute.org/

[9]http://www.washingtontimes.com/multimedia/image/indonesia-terrorjpg/

at the top of our list is Aceh, located at the north west corner of Indonesia. It is worth mentioning here that Aceh was completely devastated by the 2004 Indian Ocean Tsunami and is still recovering from its effects. Aceh is also one of the least economically developed provinces of Indonesia. We believe that due to the lack of economic advancement in Aceh, the level of Internet penetration in Aceh is fairly small and not many people from Aceh are active tweeters. This may explain the reason for Aceh not showing up among our list of top radically active provinces.

Chapter 6

# ON SOCIAL NETWORK FIREWALL SELECTION

Over the past few years, particularly with the boom of the Online Social Networks (OSNs), a considerable amount of research interest has developed in problems pertaining to the field of social networks - such as, problems of Influence Maximization [78], [79], Influence Blocking Maximization[80], [81], social network community detection [82] among others. In particular, such problems in a two-player setting has gained much interest in the recent past. This is because it is quite evident from common sense that one's decision to adopt a new technology or product is often made under the influence of one's friends and family. Besides, in the real world, there are always more than one competing technology or product to choose from. Each competing company (or player) in the market would like to win over as many loyal followers as possible but at the same time, she would also like to try to "contain" the spread of the other player's product. Besides, each person (represented as a node) as an individual in the social network can pose different weightage to the company - for e.g., a person's age, social status, educational status, economic status might make a person more desirable from a company's point of view. As a result, in such scenarios, it is more prudent to consider that each node in the social network has a *weight* associated with it. Also, in order to win over a population, a company might need to spend money on incentivization. Certainly, a company would like to spend as little as possible but try to reap as much benefit as possible. We can also think amount the weight of an individual from a company's point of view as the amount of incentivization required by that individual so as to become loyal to the company.

Motivated by such considerations, in this work, we consider the problem of *weighted Segregating Vertex Set problem (wSVS)*. In this problem, we are given a weighted undirected

social network graph and we consider that there are two players - A and B who are competing against each other. Let us consider that player A has already selected a subset of the population, which we refer to as the *seedset* of player A. This means that player A has already won over the loyalty of these people and as a result these people are unavailable to the second player i.e., player B. Now, player B would like to *contain* the *possible* spread of player A's influence by selecting a *firewall* of nodes from the network. This firewall will ensure that the spread of player A's influence is limited to only a part of the network such that the total weight of the nodes beyond the reach of player A is more that the total weight of the nodes within the reach of player A. This will in turn mean that even if player A is able to *influence* all the people within the reach of her seed set, she would never be able to conquer half or more than half the social network - this will give player B a reasonable chance to have a strong hold in this social network. However, from player B's perspective, she would like to construct this firewall with the minimum investment for incentivization. Thus, the problem has a definite flavor of graph vertex cut. The wSVS problem is formally defined later.

It may be noted that the wSVS problem is similar to an extent to the Influence Blocking Maximization (IBM) problem. But, wSVS is significantly different from IBM problem because in IBM problem, there is no requirement that the influence of the first player needs to be contained to less than half the entire network. Also, as a concrete real-world example competing players scenario for wSVS problem is as follows - consider two competing forces, A and B, such as two house builders or two companies manufacturing some heavy products such as cars or expensive electronics. It is evident that an individual having made expensive investments in such items is extremely unlikely to invest again in recent future. Now, if company A starts marketing while company B realizes that its product can only be ready in about half a year or so, company B would like to stop customers from buying from company A in the meantime. Such scenarios are exactly captured by the wSVS problem.

Even though the primary setting of the wSVS problem is in the domain of social networks, it may be noted that in abstraction, the problem can be easily imagined in the domains of damage control or epidemic control. Consider, for example, that a part of a population has become infected by a contagious disease. A primary prevention method by health service officials could be to try to vaccinate such a firewall of people such that more than half of the population would be safe-guarded against the epidemic. Similarly, in a distributed data storage network, if a part of the network becomes compromised, it may be prudent to protect such a firewall such that more than half of the network is protected. Although it might appear that these scenarios are completely different from the social network scenarios described earlier, in abstraction the underlying problem is the same.

The rest of the chapter is organized as follows - in section 6.2, we formally define the weighted Segregating Vertex Set (wSVS) problem as well as provide a formal proof of the hardness of the wSVS problem; in section 6.3, we provide an optimal solution to the wSVS problem using mixed integer linear program formulation as well as a heuristic solution to solve the wSVS problem in polynomial time. In section 6.4, we demonstrate the efficacy of our heuristic solution through detailed experimentation of three families of network namely Barabasi-Albert graph, Erdos-Renyi graph and Watts-Strogatz graph. In all the test cases, our heuristic provides near optimal solution in a fraction of the time necessary for obtaining the optimal solution.

## 6.1 Related Works

The research community in recent times has seen a heightened level of interest in *social computing* or *social network* problems. In the Influence Maximization (IM) problem [78], [83], given a network, a player wants to incentivize a given number of nodes so as to obtain the maximum number of loyalists in the network. The natural generalization of this problem

is to extend the setting to a multi-player scenario [79], where there might be multiple players trying to capture a given market. A number of different models of propagation of influence through a social network has been proposed - these include probabilistic influence propagation [78] as well as deterministic influence propagation [84]. [85] is a variation of the IM problem in a two player setting where the first player has already selected a seed set and the goal of the second player is to select a subset (of minimum cardinality) of nodes from the remaining population such that after the influence from the seed set of both the players propagate, the expected number of nodes influenced by second player is strictly greater than that by the first player.

Influence Blocking Maximization (IBM) is the problem where the second player attempts to stall the influence propagation of the first player (under a budget constraint) to as high an extent as possible through strategic selection of a seed set that could initiate influence propagation of its own. [80] proposes a solution technique for the IBM problem under competitive linear threshold (CLT) model. [86], [87] are works on variations of the IBM problem from a game theoretic perspective.

Another line of research involves the propagation of negative influence, contagious diseases and so on. [88] studies the problem of minimizing the influence of negative information. In [89], given a graph where a node has been marked to be the ‘source’, the goal is to obtain a cut minimizing the number of nodes on the partition containing the source, such that the capacity of the cut does not exceed a pre-determined budget.

Although all these studies delve into different aspects of social network problems and there are numerous studies on the Set Partition Problem [90],[91], to the best of our knowledge none of them focus on problems related to the wSVS problem where there is a strict constraint that the weighted sum of the nodes reachable from the first player is to be restricted to less than half of the total weighted sum of all the nodes in the network. Here, by reachability, we

generalize to any model of influence propagation. This means that irrespective of the model of propagation considered, the total weighted influence of the first player must be restricted to less than half of the total weight of the entire network.

## 6.2 Problem Formulation

In this section, we provide a formal statement of the weighted Segregating Vertex Set (wSVS) problem. The wSVS problem is defined as follows:

Given a weighted undirected graph $G$ and a subset $R \subset V(G)$, find a least weighted vertex separator $C$ ($C \subset V(G) \setminus R$) such that $C$ divides the graph G into two components (say, $P$ and $Q$), where (i) $R \subseteq P$ and (ii) $weight(P) < weight(Q) + weight(C)$.

It may be noted that $R$ in this formulation represents the seed set of the first player (referred to as $seedset_A$) in discussion earlier. For a subset of nodes $V' \subset V(G)$, we define $weight(V') = \sum_{v \in V'} w_v$, where $w_v$ is the weight of the node $v$. For the decision version of the wSVS problem, the question is as follows: Is there a vertex separator $C$ of weight at most $B$, such that

$$weight(P) < weight(Q) + weight(C) \text{..... (i)}$$

We assume that $\sum_{v \in seedset_A} w_v < \sum_{v \in V(G) \setminus seedset_A} w_v$ i.e., if the second player selects all the remaining nodes after selection of the nodes by the first player, constraint (i) will certainly be satisfied.

**Theorem 6.1.** *The wSVS problem is NP-complete.*

*Proof.* Given an instance of the wSVS problem and a solution to the problem (i.e., the vertex separator $C$), it is easy to verify in polynomial time whether $C$ indeed provides a feasible solution. Accordingly, wSVS is in NP. We prove that the wSVS problem is NP-complete by reducing the Set Partition Problem, a well-known NP-complete problem, to it.

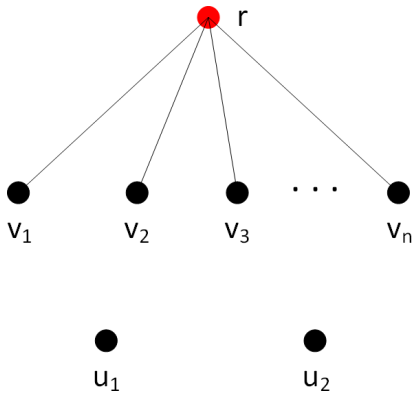



Figure 19: Construction for hardness proof of wSVS problem

*Set Partition Problem:* An instance of the Set Partition problem is made up of a set of integer numbers $S$, and it asks the following question: Is there a partition of the set $S$ into two subsets $A$ and $\bar{A}$ ($\bar{A} = S \setminus A$), such that the sum of the integers in the two sets $A$ and $\bar{A}$ are equal (i.e., $\sum_{x \in A} x = \sum_{x \in \bar{A}} x$).

Given an instance of the set partition problem, we construct an instance of the wSVS problem as shown in Fig. 19:

Let us enumerate the numbers in $S$ as $s_1, s_2, \ldots, s_n$. For each number $s_i \in S$, we create a node $v_i$ of weight $w_{v_i} = value\ of\ s_i$. For e.g., if the enumeration of $S$ is as follows $5, 3, 6, \ldots$, then $w_{v_1} = 5, w_{v_2} = 3, w_{v_3} = 6, \ldots$. For notational purpose, let us denote $V = \{v_1, v_2, \ldots, v_n\}$. We add a single red node $r$ (forming the seed set of the first player) and add undirected edges $(r, v_i), 1 \leq i \leq n$. Also, we add two disjoint nodes $u_1$ and $u_2$, where $w_r = w_{u_1} = w_{u_2} = 1$. Let, $T = \sum_{i \in S} i$ i.e., sum of all the elements of $S$. Let $B = \frac{T}{2}$. We next provide both directions of the NP-hardness proof.

**I**f case: If there is a partition of $S$ into $A$ and $\bar{A}$, then $C = A$. So, $weight(C) = B = \frac{T}{2}$ and $P = \{r\} \cup \bar{A}$ and $Q = \{u_1, u_2\}$ and thus condition (i) is satisfied.

**O**nly if case: If there is a yes answer for the wSVS problem, let $C$ be the corresponding vertex separator where $weight(C) \leq \frac{T}{2}$ and condition (i) is satisfied.

Now, if $weight(C) < \frac{T}{2}$, then $weight(V \setminus C) > \frac{T}{2}$. Since, all the weights are integers, we can say that

$$weight(V \setminus C) - weight(C) \geq 1 = weight(\{u_1, u_2\}) - w_r$$
$$\implies weight(V \setminus C) + w_r \geq weight(C) + weight(\{u_1, u_2\})$$

which violates condition (i). This implies that $weight(C) = \frac{T}{2} = weight(V \setminus C)$ and a partition of $S$ exists. □

## 6.3 Solutions for the wSVS problem

In this section, we provide optimal and heuristic solutions for the wSVS problem.

### 6.3.1 Optimal Solution

We provide an optimal solution for the wSVS problem using Mixed Integer Linear Program (MILP) formulation. Given a graph $G$, we define the following variables:

For each node $v \in V(G)$:

$X_v = 1$, if node $v$ belongs to $P$ and 0 otherwise,

$Y_v = 1$, if node $v$ belongs to $Q$ and 0 otherwise.

Let $V(G)$ and $E(G)$ denote the vertex set and edge set of $G$ respectively. The MILP can now be written as:

$$min \sum_{v \in V} w_v \times (1 - X_v - Y_v)$$
$$X_u + Y_v \leq 1 \quad \forall (u, v) \in E(G) \tag{6.1}$$

$$X_v + Y_v \leq 1 \quad \forall v \in V(G) \tag{6.2}$$

$$X_v = 1 \quad \forall v \in R \tag{6.3}$$

$$\sum_v (w_v \times X_v) < \sum_v (w_v \times Y_v) + \sum_v (w_v \times (1 - X_v - Y_v)) \tag{6.4}$$

$$X_v \ \in \ \{0, 1\}; Y_v \ \geq \ 0;$$

The objective function implies that we want to minimize the total weight of the nodes in the separator $C$ (or equivalently maximize the total weight of the nodes which are as assigned to components $P$ and $Q$). Constraint (6.1) implies that there can be no edge between $P$ and $Q$. Constraint (6.2) implies that $P \cap Q = \emptyset$. Constraint (6.3) implies that $R \subseteq P$ or equivalently $seedset_A \subseteq P$. Finally, constraint (6.4) implies that $\sum_{v \in P} w_v < \sum_{v \in Q} w_v + \sum_{v \in C} w_v$.

### 6.3.2 Heuristic Solution

Since, solving MILP can be NP-hard, we provide a heuristic solution by solving the Linear Program (LP) with relaxed integrality constraints of the MILP formulation given in section 6.3.1 and then using the output of the LP in order to obtain the final firewall.

**Algorithm 11:** Heuristic algorithm for solving wSVS problem

```
Solve the relaxed linear program formulation for the wSVS problem;
Initialize C as the empty-set;
while total weight of nodes reachable from seedset_A is greater than or equal to total
  weight of nodes unreachable from seedset_A do
    Add node v to C where v has the highest value of 1 − X_v − Y_v among all
      v' ∈ V(G) \ C;
    breaking ties randomly;
return C;
```

#### 6.3.2.1 Description

The output of the relaxed LP formulation for the MILP formulation given in section 6.3.1 gives fractional values (referred to as *l*p values by us) to the nodes of the input social network graph $G$. The MILP formulation very evidently has no properties such as half integrality or total-unimodularity. So, our heuristic solution performs rounding of the *l*p values for the nodes. Since, the heuristic will select some nodes from $V(G) \setminus seedset_A$ for the final output set $C$, constraints $1-3$ of the MILP formulation are automatically satisfied. Besides, we are explicitly checking whether constraint (6.4) is satisfied in the condition of the **while** loop of line $3$ and so constraint (6.4) is also always satisfied. The heuristic selects node in non-increasing order of the values $(1 - X_v - Y_v), \forall v \in V(G)$ assigned by the solution of the relaxed LP formulation, breaking ties randomly. The intuition behind this is that higher the value $(1 - X_v - Y_v)$ for a node $v$, the greater fraction of the node $v$ has been used by the linear program solution in its final solution. Since, we can not use a fraction of a node as the final solution of wSVS problem, the heuristic includes the entire node in its solution set $C$. The efficacy of this simple algorithm is proven empirically through our experimentation provided in section 3.3.

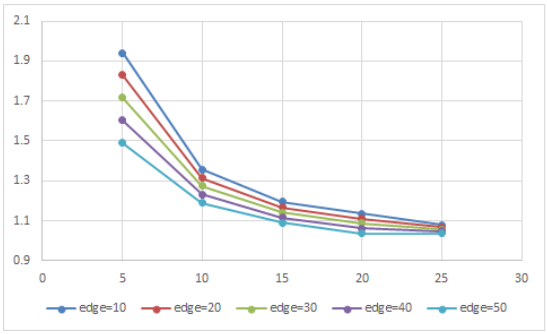
#### 6.3.2.2 Time Complexity

Step 1 of solving the relaxed LP formulation of the MILP formulation given in section 6.3.1 takes polynomial time. The condition for the **while** loop of steps $3-5$ can be computed through a graph traversal algorithm such as depth-first search or breadth-first search which takes $O(|V(G)| + |E(G)|)$ time. We can sort and store the $1 - X_v - Y_v$ values as a pre-computation step for efficient computation of step $4$. A standard sorting algorithm on $O(n)$

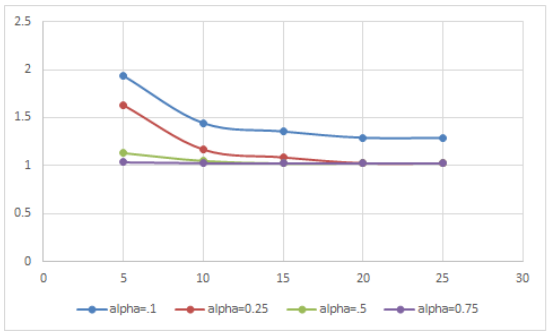
nodes takes $O(n\ log\ n)$ time. The **while** loop can be executed for a maximum of $V(G) - |seedset_A|$ which is $O(n)$. Hence, Algorithm 11 runs in polynomial time in $|V(G)|$.

## 6.4 Experimental Results and Discussions

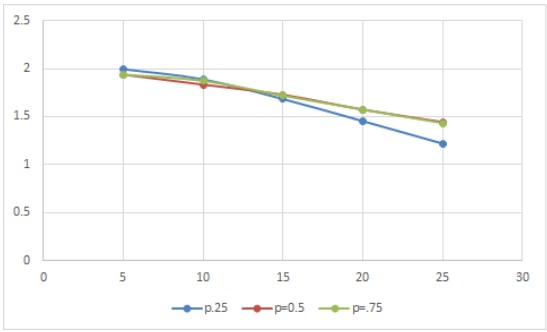
In this section we present results of our experimentations to prove the effectiveness of our simple heuristic algorithm. For this, on one hand, we consider three families of graphs namely - Barabasi-Albert graph [92], Erdos-Renyi graph [93] and Watts-Strogatz graph [94] on 100 nodes and different parameters relevant for the particular type of graph. And on the other hand, we conisder an ego-Facebook real world dataset (freely available for download from https://snap.stanford.edu/data/) consisting of $4039$ nodes and $88,234$ edges. For generating data for the three graph families, we have used the NetworkX python library and because these are random graphs, we have experimented with $500$ instances for each set of parameters. Barabasi-Albert graphs are random scale-free networks generated using a preferential attachment algorithm. This means that a graph of $n$ nodes is constructed through the process of attaching new nodes each with a specified number of edges that are preferentially attached to existing nodes with high degree. The reason we have chosen Barabasi-Albert graph as one of the families of graphs for our experiment is that such scale-free networks are frequently observed in different social networks. For Barabasi-Albert graphs, in the context of wSVS problem, the parameters that we have experimented with are - (i) *m* i.e., the number of edges to attach from a new node to existing nodes and it has been varied from $10$ to $50$ in steps of $10$ as well as (ii) the size of seed set of player A and it has been varied from $5$ to $25$ in steps of $5$. Erdos-Renyi graphs are a family of random graphs. We have considered Erdos-Renyi graphs as baseline graph family. The parameters of Erdos-Renyi graphs, in the context of wSVS problem, that we have experimented with are - (i) *p* i.e., the probability that each edge exists and it



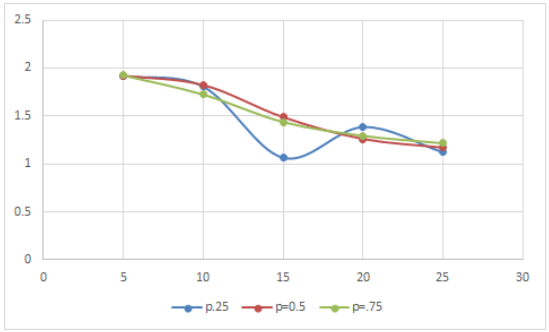

(a) Results for Barabasi-Albert Network



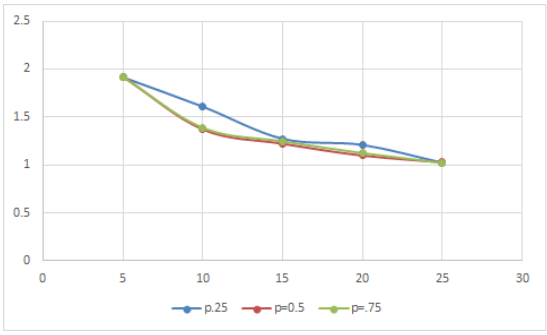

(b) Results for Erdos-Renyi Network



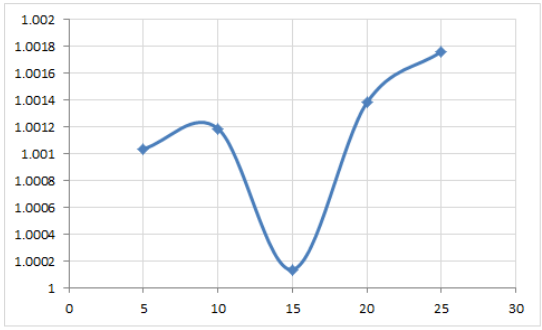

(c) Results for Watts-Strogatz graph for $k = 4$




(d) Results for Watts-Strogatz graph for $k = 10$




(e) Results for Watts-Strogatz graph for $k = 16$




(f) Results for Facebook data

Figure 20: Experimental results for Barabasi-Albert Network, Erdos-Renyi Network, Watts-Strogatz Network and Facebook graph for different parameters

has been given values of $0.1, 0.25, 0.5, 0.75$ as well as (ii) the size of the seed set of player A and it has been varied from $5$ to $25$ in steps of $5$. Finally, Watts-Strogatz graphs is a family of graphs with *s*mall-world properties, which include high clustering properties and short average path lengths. Such networks are also seen in social networks. For the Watts-Strogatz graphs, the parameters that we have experimented with in the context of the wSVS problem are - (i) $k$ i.e., each node is connected to $k$ nearest neighbors in ring topology and it has been given values of $4, 10, 16$, (ii) $p$ i.e., the probability of rewiring each edge and it has been given values of $0.25, 0.5, 0.75$, and (iii) size of the seed set of player A and it has been varied from $5$ to $25$ in steps of $5$. For the Facebook dataset, we have used the data as is and have considered the entire network graph with all the nodes and all the edges from all the egonets combined. For all the datasets, the degree of each node in the graph is assigned as its weight. With these datasets in hand, we have computed the optimal solution for the wSVS problem by solving the MILP formulation as provided in section 6.3.1 by using CPLEX Optimization Studio 12.5. The heuristic solution is implemented in Java and run for these datasets on a Windows 7 Intel core i7 laptop. The results are plotted in Fig. 20, where for each graph, we plot along x-axis the percentage of total number of nodes selected by the first player as her seed set - for e.g., at $x = 15$, we plot the results when the first player has selected $15\%$ of the total number of nodes as her seed set. And along y-axis, we plot the ratio of the weight of the seed set selected by the heuristic solution to the weight of the seed set selected by the optimal solution. In all of our experiments, the heuristic has obtained a solution value within a factor of $2$ of the optimal solution value but in lesser time compared to that required to compute the optimal solution.

Chapter 7

# WINNING WITH MINIMUM INVESTMENT UNDER SEPARATED THRESHOLD MODEL (WMI-LT)

It has been frequently observed in different studies in social sciences and economics that people are more often than not influenced by recommendations of friends and family regarding decision making about adopting a new product or technology. As a result, over the past several years, there has been numerous studies in the domain of influence propagation and influence maximization problems in social networks [95]–[101]. The major goal of the classical problem of influence maximization is the identification of *k most influential nodes* in a network. In order to initiate a word-of-mouth positive influence propagation about a new product, the product manufacturer might want to identify the $k$ most influential nodes in the network, such that she can incentivize these influential people to buy this new product by, say, providing free samples to them. There might be a budget on the amount of money that she is able to spend on advertisement and incentivization - hence, is the need to identify the *most influential* people in the network and provide free samples to only them. The value of the parameter $k$ is determined by the size of the available advertising budget.

The studies of influence propagation can be broadly classified into the following three categories based on the influence propagation models used.

- Class I: Non-adversarial
- Class II: Adversarial with passive adversary
- Class III Adversarial with active adversary

In most of the influence propagation models, influence propagates in a step-by-step fashion and as such there is a notion of time step (or propagation step) involved. The expected

number of nodes influenced at the end of time step $D$ is at most the expected number of nodes influenced at the end of time step $D+1$. In other words, expected number of nodes influenced at the end of time step $D$ is a *non-decreasing* function of $D$.

Studies in Class I [100] focusing on *non-adversarial environment* consider that there is only one manufacturer (player) which is attempting to influence the nodes of a social network to buy her new product. This can be perceived as the following - all the nodes in the social network are initially white i.e., they have not yet adopted the new product but is open to the idea of doing so. The manufacturer (player) wants to color a few nodes red by providing them with some incentives. Next, based on the particular model of influence propagation, the initial red nodes will gradually turn some other white nodes into red. The player wants to maximize the number of red nodes at the end of the propagation process.

Studies in Class II and III focus on the more realistic *adversarial environment* by considering that there are multiple players and each of them is attempting to sell their competing products or innovations and capture as big a share of an emerging market as possible. Given that some nodes are already colored red, studies in Class II consider the problem of which $k$ white nodes should be colored blue, so that this set of nodes will have the largest impact in preventing the white nodes from turning red. Hence, Class II scenario can be viewed as a *passive adversary* setting, because its goal is to prevent white-to-red conversion, and it is not engaged in white-to-blue conversion. Finally, studies in Class III consider *active adversary* scenario, where the red agent is actively engaged in white-to-red conversion, while the blue agent is also actively engaged in white-to-blue conversion.

In [96], the authors study a two-player problem belonging to Class III, where the goal of the second player is to maximize her own influence given that the first player has already selected a set of $k$ initial nodes. In [85], the authors consider the 'Winning with Minimum Investment (WMI)' problem which also belongs to Class III category. Here, two manufacturers

(players) are trying to sell their competing products by incentivizing and thereby influencing the nodes of a social network. The goal of both the players is *to have a market share that is larger than its competition*. The authors consider the scenario where the first player ($P_1$) has already chosen the $k$ nodes to have a large influence (coverage) on the social network. The second player is aware of the first player's choice and the goal of the second player ($P_2$) is to *identify a smallest set of nodes (excluding the ones already chosen by the first player) so that the number of nodes influenced by the second player will be larger than the number of nodes influenced by the first player within $D$ time steps*. In other words, the objective of the second problem is to *minimize the incentivization cost subject to the constraint that the coverage of the second player is larger than the coverage of the first player within $D$ time steps*. Both [96] and [85] consider two models of influence propagation that were introduced in [96] - namely the Distance-based Model and the Wave-propagation Model which are generalizations of the Independent Cascade (IC) model [78] . In this dissertation, we study the WMI problem in a different model of propagation namely the Separated Threshold Model (SepT) [102] which is a generalization of the Linear Threshold (LT) model [78]. We find that similar results as in [85] hold even under the SepT model. In Section 7.1, we discuss the previous studies related to our work. In Section 7.2, we provide the formal definition of the WMI problem under SepT model of propagation (WMI-LT). In Section 7.4, we prove that the WMI problem is also hard under the SepT model. In Section 7.3, we provide an equivalent random process to SepT model and in Section 7.5 we provide an approximation algorithm for solving the problem. Finally, in Section 7.6, we present our experimental results.

## 7.1 Related Works

Kempe, Kleinberg and Tardos in their seminal work [100] initiated a wave of interest in the research community for problems in the domain of influence propagation and maximization by proposing new models derived from mathematical sociology and interacting particle systems. They formulated the *influence maximization problem* and provided approximation algorithms for the same by utilizing the *submodularity* property of the objective functions. In addition to that, through experiments conducted on large collaboration networks, they showed that their greedy approximation algorithm performed significantly better than node selection heuristics based on *degree centrality* and *distance centrality*. [100] sparked the beginning of much research in problems belonging to Class I scenario. Additionally, because the greedy algorithm of [100] is computation-intensive, much research has been conducted in improving its scalability. Chen et. al. in [97] improved the greedy algorithm with the help of a *degree discount* heuristic for improving influence spread. Mathioudakis et. al. in [101] introduced the SPINE algorithm which when used as a pre-processing step for the influence maximization problem, significant speedup is possible without affecting the accuracy. [103] presents a "lazy-forward" optimization in selecting new seeds. Utilizing this strategy, the authors show that it is possible to greatly reduce the number of evaluations of the influence spread of nodes which forms a key point where the original greedy algorithm suffers. The authors demonstrate experimentally that the resultant speedup is as high as 700 times. [104] devise a heuristic algorithm that is scalable to millions of nodes and edges. [105] presents the first scalable influence maximization algorithm for, in particular, the linear threshold model of influence propagation. [106] proposes SIMPATH which is an efficient algorithm for influence maximization under the linear threshold model employing several effective optimizations.

Several variations of the original Influence Maximization problem formulation as well as the computation model have been studied in the research community. In [107], the authors consider the case that a user (a node in the social network) may not necessarily adopt the influence or product herself, but may convey positive feedback about the product to her friends. The authors in [107], thus, study an "adoption maximization" problem instead of "influence maximization" problem. Similar considerations are also made by [108] where the authors propose a diffusion model which is a generalization of the SIR model and is relevant for the diffusion of information through a micro-finance loan network in a village. In [109], the authors have proposed a new problem which they refer to as the Seed Minimization with Probabilistic Coverage Guarantee (SM-PCG). The SM-PCG problem is as follows - given a social network modeled as a directed social graph $G = (V, E)$, where $V$ is the set of $n$ nodes representing individuals in a social network and $E$ is the set of directed edges representing influence relationships between pairs of individuals. Each edge $(u, v) \in E$ is associated with an influence probability $p_{u,v}$ which is the probability that node $u$ activated node $v$ after $u$ is activated. The influence diffusion process in the social network graph $G$ follows the IC model. Given a target set $U \subseteq V$, let $Inf_U(S)$ be the random variable denoting the number of active nodes in $U$ after the diffusion process starting from the seed set $S$ ends. Let $Inf(S)$ refer to the influence coverage of seed set $S$ (for target set $U$). The optimization problem considered in this work is to find a seed set $S$ of minimum size such that the influence coverage of $S$ is at least a required threshold with a required probability guarantee. [110] studies the problem of identifying influential and susceptible members of social networks through the usage of in vivo randomized experimentation to identify influence and susceptibility in networks and thereby avoiding the biases inherent in traditional estimates of social contagion. Their study combine analysis of influence and susceptibility together with network structure.

Several studies have also been conducted for Class II problems (adversarial with passive adversary). [111] studies the problem of identification of *blockers*, meaning the nodes that can most effectively block the spread of a dynamic process through a social network. The authors suggest that simple local heuristics such as the node degree are good indicators of its effectiveness as a blocker. [89] study the problem of minimizing the number of nodes reachable from the nodes selected by the first player when the second player has a budget on the amount of available incentives. There has been considerable research effort targeting the blocker identification problem in public health community as well as the fields of epidemiology, disaster control, military containment.

The WMI problem studied in [85] as well as this dissertation belongs to Class III (adversarial with active adversary). There are only limited studies on problems belonging to Class III. One of the earliest studies of a Class III problem was conducted in [95]. The authors of [95] propose a mathematical model for diffusion of multiple innovations in a network. They use game theoretic framework and propose an approximation algorithm with an approximation ratio of $(1 - 1/e)$ for computing the best response to an opponent's strategy. An algorithmic framework for studying a Class III problem was proposed in [96]. The authors of [96] extend the Influence Maximization problem studied in [100] from the Class I scenario to the Class III scenario. They study how a follower (i.e., the player who entered the market after the first player) can maximize her influence in the network under a budget constraint, and under the condition that the first player has already incentivized and thereby influenced some individuals (nodes in the network). They prove the problem to be NP-complete and provide an approximation algorithm with guaranteed performance bound of being within 63% of the optimal.

## 7.2 Problem Formulation

In [85], the authors consider the problem called 'Winning with Minimum Investment' (WMI) problem. The WMI problem can be stated as follows: Given a diffusion model and the information that a subset of nodes $I_A$ has already adopted innovation A marketed by player $P_1$, what is the fewest number of nodes that player $P_2$ (marketing innovation B) should target so that by the end of $D$ time steps, the number of nodes that adopt innovation B will exceed the number of nodes that adopt innovation A? If $\sigma_1(I_A, I_B, D)$ and $\sigma_2(I_A, I_B, D)$ denote the expected number of nodes that adopt innovations A and B respectively within $D$ time steps, the objective of the WMI problem is to

$$\text{minimize } |I_B|$$
$$\text{subject to } \sigma_2(I_A, I_B, D) > \sigma_1(I_A, I_B, D)$$

In [85], even though the authors do consider adversarial scenario but the propagation models considered are Wave propagation model and Distance-based models [96] (which stem from the Independent Cascade model [100]) which do not consider the realistic case that each individual (node in a social network) might not have the same inclination or natural propensity towards two separate competing sources of influence. Such a scenario is modeled by the "Separated Threshold Model" (SepT) [102] and also the "Competitive Linear Threshold Model" (CLT) considered in [80]. These two models are almost identical sans they use different tie breaking rules. We next describe the difference between the two models. Consider an adversarial setting in which there are two competing sources of influence namely A and B which are trying to capture a new market represented by a graph $G = (V, E)$. Each edge $(u, v) \in E$ is assigned a real-valued weight representing each technology $w^A_{u,v}, w^B_{u,v} \in [0, 1]$, such that $\sum_u w^A_{u,v}, \sum_u w^B_{u,v} \in [0, 1]$ which represents node $u's$ impact on $v$. Let $I^0_A, I^0_B \subseteq V$ (where $I^0_A \cap I^0_B = \emptyset$) represent the initial A-active and B-active nodes respectively. At time step $t = 0$,

each node $v \in V$ selects two thresholds $\theta_v^A, \theta_v^B \in_R [0,1]$. Let $I_A^{t-1}, I_B^{t-1}$ represent the sets of A-active and B-active nodes at time step $t$. In a time step $t$, for an inactive node $v \in V$ if $\sum_{u \in I_A^{t-1}} w_{u,v}^A \geq \theta_v^A$, $v$ will become A-active and $v$ will become B-active if $\sum_{u \in I_B^{t-1}} w_{u,v}^B \geq \theta_v^B$. If both thresholds are exceeded during the time step $t$ for the node $v$, then $v$ adopts a cascade uniformly at random in the SepT model. Whereas, in case of a tie, $v$ becomes A-active in the CLT model - this makes sense when we consider that the two propagations are positive ($P_2$ *i.e.* B) and negative rumors ($P_1$ *i.e.* A) and captures the negativity bias phenomenon which is well studied in social psychology .

## 7.3 Active Edge Equivalent Model

In this section, we describe an active edge model which is an equivalent random process of the influence propagation using SepT model. The random process is the same as described in [80] but the analysis is a little different because of the difference in the tie breaking rule used in the CLT model used in [80] and the tie breaking rule in SepT model. We include the details of the random process here for comprehensiveness.

We construct a *random live-path graph* $G_X$ from the given graph $G = (V, E)$ as follows. For each node $v \in V$, we randomly pick one A-type in-edge $(u, v)$ with probability $w_{u,v}^A$, and with probability $1 - \sum_{u \in V} w_{u,v}^A$ no A-type in-edge is selected; similarly, we also randomly pick one B-type in-edge $(u, v)$ with probability $w_{u,v}^B$ and with probability $1 - \sum_{v \in V} w_{u,v}^B$ no B-type in-edge is selected. Let us denote the subgraph of $G_X$ consisting of only B-type edges by $G^B$, and let us denote the subgraph of $G_X$ consisting of only A-type edges by $G^A$. Given $I_A^0, I_B^0 \subseteq V$ which represent the initial A-active and B-active nodes respectively, let us define $d_{G^B}(I_B^0)$ be the shortest graph distance from any node in $I_B^0$ to $v$ only through the B-type edges, and $d_{G^A}(I_A^0)$ be the shortest graph distance from any node in $I_A^0$

to $v$ only through the A-type edges. This distance could be $\infty$ if no such path exists. As a result, in the random live-path graph, we say that a node $v$ is B-active if $d_{G^B}(I_B^0) < \infty$ and $d_{G^B}(I_B^0) < d_{G^A}(I_A^0)$, and $v$ is A-active if $d_{G^A}(I_A^0) < \infty$ and $d_{G^A}(I_A^0) < d_{G^B}(I_B^0)$. If for some node $v$, $d_{G^B}(I_B^0) \neq \infty, d_{G^A}(I_A^0) \neq \infty, d_{G^B}(I_B^0) = d_{G^A}(I_A^0)$, then there is a fifty percent chance of $v$ becoming A-active and a fifty percent chance of $v$ becoming B-active. The following lemma shows that the A and B type activation sets generated by the above random process is equivalent to the corresponding one generated by the SepT model.

**Lemma 7.1.** *For a given initial A-active and B-active sets $I_A^0$ and $I_B^0$, the distribution over A-active and B-active sets is identical in the following two definitions.*

1. *distribution obtained by running SepT process,*
2. *distribution obtained from reachability defined above in the live-path graph.*

*Proof.* The activation process under the SepT model consists of several iterations. In each iteration, some nodes change from inactive (white) to B-active (blue) or A-active (red). As mentioned earlier, $I_A^t, I_B^t$ represent the set of nodes which are A-active and B-active at the end of iteration $t$. At time $t$, let us consider an inactive node $v \notin I_A^t \cup A_B^t$. Now, the probability of $v$ becoming B-active in iteration $t+1$ equals the chance that the B-type edge weights in $I_B^t \setminus I_B^{t-1}$ push it over $\theta_v^B$ while the A-type weights is still less than $\theta_v^A$ or both thresholds are exceeded and $v$ becomes B-active with probability $0.5$. The above probability under the condition that for the node $v$, neither A-type nor B-type threshold is exceeded already by iteration $t$ is:

$$\frac{(\sum_{u\in I_B^t\setminus I_B^{t-1}} w_{u,v}^B)(1-\sum_{u\in I_A^t\setminus I_A^{t-1}} w_{u,v}^A)+\frac{1}{2}(\sum_{u\in I_B^t\setminus I_B^{t-1}} w_{u,v}^B)(\sum_{u\in I_A^t\setminus I_A^{t-1}} w_{u,v}^A)}{(1-\sum_{u\in I_A^{t-1}} w_{u,v}^A)(1-\sum_{u\in I_B^{t-1}} w_{u,v}^B)}$$

Again, we can obtain the probability that a node $v$ becomes A-active in iteration $t+1$ given that $v$ is inactive from iteration $0$ to $t$. The probability is:

$$\frac{(\sum_{u\in I_A^t\setminus I_A^{t-1}} w_{u,v}^A)(1-\sum_{u\in I_B^t\setminus I_B^{t-1}} w_{u,v}^B)+\frac{1}{2}(\sum_{u\in I_A^t\setminus I_A^{t-1}} w_{u,v}^A)(\sum_{u\in I_B^t\setminus I_B^{t-1}} w_{u,v}^B)}{(1-\sum_{u\in I_A^{t-1}} w_{u,v}^A)(1-\sum_{u\in I_B^{t-1}} w_{u,v}^B)}$$

Next, we consider the above discussed probability when using the random live-path graph. Initially we have A-active and B-active node set as $I_A^0$ and $I_B^0$ respectively - let us refer to them as $J_A^0$ and $J_B^0$. For each time step $t = 1, 2, \ldots$, we define $J_A^t$ to be the set of nodes containing any $v \notin J_A^{t-1} \cup J_B^{t-1}$ such that $v$ has one in-edge from some node in $J_A^{t-1}$; we define $J_B^t$ to be the set of nodes containing any $v \notin J_A^{t-1} \cup J_B^{t-1}$ such that $v$ has one in-edge from some node in $J_B^{t-1}$.

By the definition of the random live-path graph, the probability that a node $v$ is in $J_B^{t+1} \setminus J_B^t$ conditioned on that $v \notin J_A^t \cup J_B^t$ is

$$\frac{(\sum_{u \in J_B^t \setminus J_B^{t-1}} w_{u,v}^B)(1-\sum_{u \in J_A^t \setminus J_A^{t-1}} w_{u,v}^A)+\frac{1}{2}(\sum_{u \in J_B^t \setminus J_B^{t-1}} w_{u,v}^B)(\sum_{u \in J_A^t \setminus J_A^{t-1}} w_{u,v}^A)}{(1-\sum_{u \in J_A^{t-1}} w_{u,v}^A)(1-\sum_{u \in J_B^{t-1}} w_{u,v}^B)}$$

Similarly, the probability that a node $v$ is in $J_A^{t+1} \setminus J_A^t$ conditioned on that $v \notin J_A^t \cup J_B^t$ is

$$\frac{(\sum_{u \in J_A^t \setminus J_A^{t-1}} w_{u,v}^A)(1-\sum_{u \in J_B^t \setminus J_B^{t-1}} w_{u,v}^B)+\frac{1}{2}(\sum_{u \in J_A^t \setminus J_A^{t-1}} w_{u,v}^A)(\sum_{u \in J_B^t \setminus J_B^{t-1}} w_{u,v}^B)}{(1-\sum_{u \in J_A^{t-1}} w_{u,v}^A)(1-\sum_{u \in J_B^{t-1}} w_{u,v}^B)}$$

Evidently, the above conditional probabilities are the same as those derived from the SepT model. Also, since $I_A^0 = J_A^0$ and $I_B^0 = J_A^0$, by induction over the iterations, we conclude that the random live-path graph model produces the same distribution over A-active and B-active sets as the SepT model. □

## 7.4 Hardness of the *W*MI-LT Problem

Next, we prove that the WMI-LT problem under the SepT model is NP-hard in lines with the hardness proof in [85]. The decision version of *W*MI-LT is *"Is there a subset $I_B$ where $|I_B| \leq M$ and $\sigma_2(I_A, I_B, D) > \sigma_1(I_A, I_B, D)$?"*

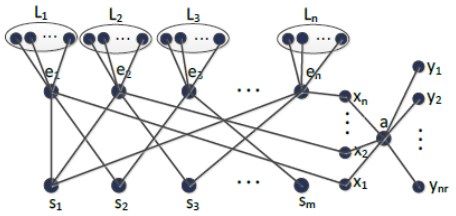



Figure 21: Graph $G = (V, E)$ of WMI-LT instance in set cover reduction

**Theorem 7.2.** *WMI-LT is NP-hard for the SepT model*

*Proof.* For the purpose of this proof, we reduce the known NP-complete Set Cover problem to WMI under the SepT model. We know that the decision version of the Set Cover problem is defined in the following way: A ground set of elements $S = \{e_1, e_2, \ldots, e_n\}$, a collection of sets $C = \{s_1, s_2, \ldots, s_m\}$ such that $s_i \subseteq S$ and a positive integer $K \leq |C|$ are given. The question is whether there exists a collection $Q \subseteq C$ that covers all the elements in $S$ and $|Q| \leq K$.

Given an instance of set cover problem, we construct an instance of the WMI-LT problem. We construct $G = (V, E)$ in the following way as shown in Figure 21. For every element $e_i \in S$, we add a node $e_i$ and for every set $s_j \in C$, we add a node $s_j$ to $V$. We add an edge $(e_i, s_j)$ to $E$ for every $e_i$ and $s_j$ if $e_i \in s_j$. Also, we add a node $a$ and nodes $x_1, \ldots, x_n$ to $V$. Then, for every $e_i$, we add edges $(a, x_i)$ and $(x_i, e_i)$ to E. Moreover, for every $e_i$ we add a set of $r$ nodes, $L_i = \{l_{i,j} | 1 \leq j \leq r\}$ to $V$ and we connect them directly to $e_i$. We identify the value of $r$ later in the proof. Finally, we add $n \times r$ additional nodes $y_1, \ldots, y_{n \times r}$ to $V$ and edges $(y_t, a), 1 \leq t \leq n \times r$ as shown in Figure 21. The weight on each edge $(u, v)$ for both $A$ and $B$ is the inverse of $degree(v)$. We consider that for each node $v$, the threshold for both player A and player B is the inverse of $degree(v)$. NP-hardness of this special case clearly implies the NP-hardness of the more general case. We assign $D = 4$ which is the diameter of the graph $G$, $M = K$ and $I_A = \{a\}$.

Now, we show that the set cover problem has a solution iff there is a set $I_B \subseteq V - I_A$ such that $|I_B| \leq M$ and $\sigma_2(I_A, I_B, D) > \sigma_1(I_A, I_B, D)$. First, we consider that there is a collection $Q \subseteq C$ that covers $S$ and $|Q| \leq K$. Then, $I_B$ includes all nodes $s_j$ corresponding to the sets in $Q$. In this case, all $e_i$ will be at distance one from $I_B$ and two from $I_A$. So, all $e_i$ and the nodes in $L_i$ will adopt $I_B$ with probability one. Moreover, the nodes $s_j \notin I_B$ are two hops away from $I_B$ while 3 hops away from $I_A$. Hence, all nodes $s_j$ will adopt $I_B$. Therefore, we have $\sigma_2(I_A, I_B, D) = m + n(1 + r)$. So, $\sigma_2(I_A, I_B, D) > \sigma_1(I_A, I_B, D)$.

Next, we show that if there is no collection $Q$ of size $K$ that covers all elements then there is no set $I_B \subseteq V - I_A$ of size $M$ where $\sigma_2(I_A, I_B, D) > \sigma_1(I_A, I_B, D)$. Considering that set cover does not have a solution, there should be at least one $e_i$ whose distance from $I_B$ cannot be one or smaller. Since the node $x_i$ connected to this $e_i$ will have probability 1 to accept A and so, this $e_i$ and consequently nodes in $L_i$ choose A. So, we have $\sigma_2(I_A, I_B, D) \leq m + (n-1)(1+r)$ and $\sigma_1(I_A, I_B, D) \geq 1 + nr + (1 + r)$. So, we choose $r$ in our instance large enough such that $r > \frac{n+m-3}{2}$. Then, we have $1 + nr + (1 + r) > m + (n-1)(1+r)$, so $\sigma_1(I_A, I_B, D) > \sigma_2(I_A, I_B, D)$ □

## 7.5 Approximation Algorithm

Having proven the hardness of approximation of WMI problem using the SepT model of propagation, we should next strive for obtaining a solution algorithm for the same. Consider the greedy algorithm provided in [85]. Let $\omega(I_A, I_B, D)$ be $(\sigma_2(I_A, I_B, D) - \sigma_1(I_A, I_B, D))$. Let $F_i$ denote the amount of increase in the value of $\omega$ when node $i$ is added to $I_B$; *i.e.* $F_i = \omega(I_A, I_B \cup \{i\}, D) - \omega(I_A, I_B, D)$. Initially, $I_B$ is the empty set. Hence, $\omega(I_A, I_B, D) \leq 0$. The algorithm executes through iterations and in each iteration, node

$i \in V - I_A$ with the maximum $F_i$ is selected. The steps of the algorithm $GWMI$ has been shown in Algorithm 12.

**Algorithm 12:** Greedy Algorithm for Winning with minimum investment (GWMI)

```
while ω(I_A, I_B, D) ≤ 0 do
    for every node i ∈ V − (I_A ∪ I_B) do
        Compute F_i;
    Select node j with maximum F_j;
    I_B = I_B ∪ {j};
return I_B;
```

**Theorem 7.3.** *GWMI has a $log\ n$ approximation ratio under SepT model.*

*Proof.* The proof follows from [85], we include it here for completeness. Let $I_B^t$ be the set of $B's$ initial adopters selected by GWMI at step $t$. Initially, $I_B$ is the empty set and $\omega(I_A, I_B^0, D) = -\sigma_1(I_A, \emptyset, D)$ In each iteration $t$, the nodes in the optimal set of $B'$s initial adopters $I_B^{opt}$, will make $\omega(I_A, I_B^{t-1} \cup I_B^{opt}, D)$ positive. We denote the size of $I_B^{opt}$ by OPT and the size of the solution of $GWMI$ by $H$. Therefore, there will be at least one node in $V - \{I_A \cup I_B^{t-1}\}$ that increases $\omega(I_A, I_B^{t-1}, D)$ by at least $\frac{|\omega(I_A, I_B^{t-1}, D)|}{OPT}$. Let, $v_t$ be the node selected by GWMI at iteration $t$. Then, $F_{v_t} \geq \frac{|\omega(I_A, I_B^{t-1}, D)|}{OPT}$. Therefore, for $t < H$, we have,

$|\omega(I_A, I_B^t, D)| \leq |\omega(I_A, I_B^{t-1}, D)| - \frac{|\omega(I_A, I_B^{t-1}, D)|}{OPT} \leq |\omega(I_A, I_B^0, D)|(1 - \frac{1}{OPT})^t$

Also, we know that $|\omega(I_A, I_B^t, D)| \leq n(1 - \frac{1}{OPT})^t \leq ne^{\frac{-t}{OPT}}$.

Since, adding a node to $I_B$ will increase $\omega(I_A, I_B, D)$ at least by one, we need to find the smallest $t$ that will make $|\omega(I_A, I_B, D)| < 1$. Then, adding one more node will make $\omega(I_A, I_B, D)$ positive. Therefore, $H \leq 1 + OPT\ ln\ n$. □

## 7.6 Experimental Results

In this section, we present our detailed experimentation on the performance of the greedy approximation algorithm, GWMI. We have performed experiments on a real dataset. We have considered the co-authorship network of scientists working on network theory and experiment, as compiled by M. Newman in May 2006 [112] because co-authorship graphs are said to be representative of social networks. This dataset has 1589 nodes and 2742 edges. Our experiments have been conducted on a Linux machine with Intel Quad core 2.66GHz processor, 8 GB memory, 3MB cache. The code for all the experiments has been written in Python and we have used NetworkX python package which is a very popular package for codes involving graphs. Because we must estimate the spread of the influence in a network through sampling, it requires us to execute our algorithm on a large number of instantiations of the social network. So, we have employed multi-threading in order to reduce the runtime.

Just as in [85], first we compare the performance of the GWMI algorithm with the results obtained from baseline heuristics which are often used to identify most influential nodes in a social network [113]. Furthermore, we also use the greedy algorithm proposed in [79] as a baseline algorithm to select seed set of the second player. We use sampling to obtain a close approximation of $\sigma_1(I_A, I_B, D)$ and $\sigma_2(I_A, I_B, D)$ with high probability.

The first baseline heuristic used is the node degree based heuristic in which the nodes are selected in decreasing order of their degrees. The second baseline heuristic used is the closeness centrality based heuristic in which the nodes are selected in the increasing order of their average distance with other nodes. Finally, we also compare with the greedy algorithm proposed in [79] which works as the following: in each iteration, the node that provides the maximum incremental increase in the total number of nodes influenced by the second player is selected. As in [85], we refer to this algorithm as the *Second Player Influence Maximization*

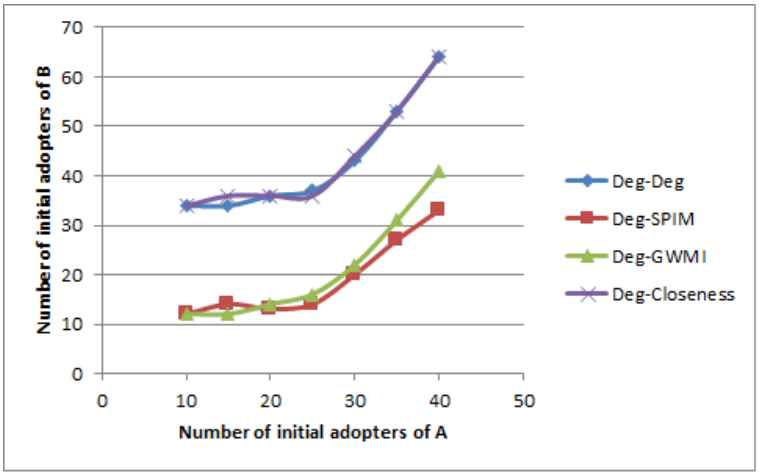
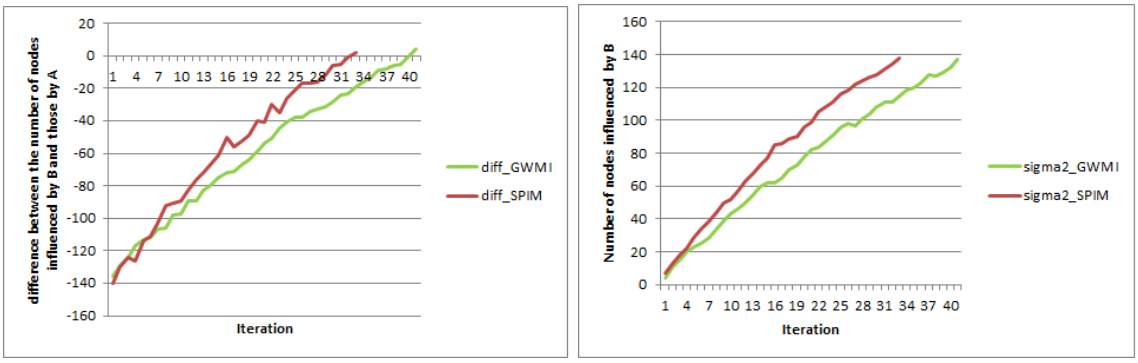



Figure 22: Figure showing the number of initial adopted of $B$ for different values of $|I_A|$

(SPIM) algorithm. In these experiments, the propagation is allowed till no more nodes get influenced. This means that we have considered the value of $D$ to be very large, for e.g., $D = n$, where $n$ is the number of nodes in the graph. In these experiments, the degree based heuristic is used to select the $k$ initial adopters for the first player $P_1$ advocating influence $A$. In our experiments, $k$ is varied from $10$ to $40$ in steps of $5$. Fig. 22 shows the plots resulting from our experiments. The 'Deg-Deg' legend shows the results when both players are using the degree based heuristic, similarly the 'Deg-GWMI' is the legend showing the results when $P_2$ uses GWMI algorithm whereas $P_1$ uses the degree based heuristic, and so on.

It is interesting to observe that for the network science dataset, for some of the values of $k$ (representing the size of the seed set of $P_1$), the GWMI algorithm selects a bigger set of initial adopters compared to the SPIM algorithm. We investigate the reason for this for the particular case where $k = 40$ because in this case the difference in the sizes of the seed sets selected by SPIM and GWMI is the biggest in our experiments and the comparison is shown in Fig. 23. It can be seen in Fig. 23a that initially the value of $\omega(I_A, I_B, D)$ is indeed higher for GWMI because by definition, GWMI selects the node $i$ which provides the highest incremental



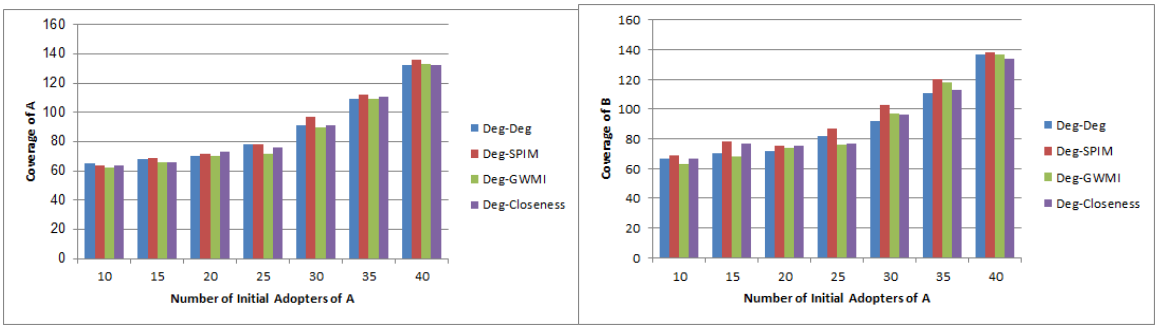

(a) Difference in the value of $\omega$ when using GWMI and SPIM

(b) Number of adopters of $B$ when using GWMI and SPIM

Figure 23: Comparison of GWMI with SPIM when first player selects $40$ nodes as initial adopters in the network science dataset.

increase ($F_i$) in the value of $\omega(I_A, I_B, D)$ when added to $I_B$. SPIM selects a different set of nodes initially than GWMI and has lower value of $\omega(I_A, I_B, D)$. However, it turns out that after a few initial iterations, SPIM beats GWMI in $\omega(I_A, I_B, D)$ where GWMI is stuck with its initial greedy selection. Fig. 23b shows that, as expected, SPIM maintains a consistently higher value of $\sigma_2(I_A, I_B, D)$ compared to GWMI because the objective of SPIM is to select the node which gives maximum incremental increase in the value of $\sigma_2(I_A, I_B, D)$ when added to $I_B$. It may be noted that in [85], the authors show that GWMI provides the smallest set of initial adopters for the WMI problem under Wave Propagation Model [79] - however they have used a different dataset than ours. Future work will involve further investigation to characterize this interesting behavior of GWMI with more datasets and different propagation models.

Next, we analyze the coverage *i.e.*the number of nodes influenced by the players. Fig. 24 presents our results. As can be seen in Fig. 24a, the coverage of $P_1$ is smaller when $P_2$ uses GWMI instead of SPIM although GWMI does not explicitly try to minimize the coverage of $P_1$. So, $P_2$ will prefer to use GWMI instead of SPIM if in addition to minimizing the number of initial adopters, $P_2$ would also like to reduce the final market share of $P_1$. Fig. 24b shows that the coverage of $P_2$ is maximum when $P_2$ uses SPIM. This is intuitive because the explicit




(a) Coverage of the first player $P_1$ (b) Coverage of the second player $P_2$

Figure 24: Coverage of the players

goal of SPIM is to maximize the coverage of $P_2$. Same results w.r.t. coverage of $P_1$ and $P_2$ have been obtained in [85] when experimenting with GWMI on a different dataset and under the Wave Propagation Model [79].

# Chapter 8

# CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

Because resource allocation problems are omni-present in the different networks, thereby influencing our lives on a daily basis, several such problems have been studied and analyzed in this dissertation. Here, networks include social networks as well as physical (infrastructure) networks such as distributed data storage networks, multi-layered interdependent networks among others.

In this dissertation, the Budget Constrained Data Distribution Problem (BCDDP) which is a problem pertaining to optimal distribution of data in a distributed data storage network has been presented. Utilizing $(\mathcal{N}, \mathcal{K})$ erasure coding, this dissertation presents a budget-constrained file distribution scheme for a data storage network where the aim is to achieve a maximum region fault-tolerant system. This ensures that, under an imposed budget, maximum number of largest connected components of the residual network has at least $\mathcal{K}$ file segments to reconstruct the file. An approximation algorithm for the problem for an arbitrary network has been presented. Besides, the efficacy of the algorithm has been demonstrated by experimentation on two real backbone networks. Also, the effects of the coding parameters $\mathcal{N}$ and $\mathcal{K}$ on storage in a distributed file storage system have been discussed. Additionally, an optimal algorithm for the all region fault tolerant file distribution system in a mesh network has been presented. Further directions of study in this domain are as follows:

- It would be interesting to study the BCDDP problem when the storage capacity of each node of the distributed data storage network is variable and could be greater than or equal to one.

- This dissertation shows that the all region fault tolerant system design problem is polynomially solvable for mesh networks, although the problem is NP-hard for general graphs. It would be interesting to study the BCDDP problem in case of such specialized or structured networks.

This dissertation also studies the relay node placement problem under budget constraint using two different metrics. It is proven that the problems using both the metrics are NP-complete and heuristic solutions for them are provided. Experimental results on synthetic data sets are also presented. Future work involves proving approximation bound for the proposed heuristics or inapproximability of the problem. Furthermore, this study has led to the development of the notion of "connectivity of a disconnected graph" or "disconnectivity", which opens up a new area of research.

- Disconnectivity of a graph (the graph does not necessarily have to be a disconnected one) can be formalized and studied in further details. Currently, two metrics of measuring connectedness of a graph is studies - namely (i) size of the largest connected component and (ii) the number of connected components. More metrics can be considered. One example is the size of smallest connected component in the graph. Disconnectivity of a graph will encompass all such metrics into a single one.

This dissertation also studies the Progressive Recovery Problem to maximize the *system utility* over the time when recovery of failed entities takes place in an interdependent network. It has been shown that the problem can be solved in polynomial time in some special case, while in others it is NP-complete. Two approximation algorithms and a heuristic have been provided to solve the problem in different cases. Additionally, Integer Programming formulations have been provided for obtaining optimal solution when the problem is

NP-complete. Experimental evaluations show that the heuristic attains near optimal solution in almost all cases. Further directions of research in this domain are the following:

- This research considers that an entity in the set of original failures is revived, it comes alive immediately. Similarly, it is also assumed that when recovery of all the entities in a minterm of the Implicative Dependency Relation of an entity takes place, the entity comes alive immediately. However, in reality, each entity may take a variable amount of time to come alive i.e., become fully functional. It would be interesting to study the progressive recovery problem under such considerations.

In this dissertation, a generic technique for recovering signals pertaining to a geographical area, such as a country, using Twitter Data has been proposed. This techniques has been applied to an Indonesian dataset of the Minerva project and a high accuracy has been observed. The goal of this work is the generation of a political Heat Map of Indonesia which highlights the Indonesian provinces with prominent radical narrative. Besides, propagation of radical or counter radical activities in a region can also be visually demonstrated through these Heat Maps. Further work in this domain could be in the following direction:

- The generic technique developed through this research can be applied to datasets from other countries or geographical areas. It would be interesting to investigate if the technique is as effective in those datasets as it is for the Indonesian dataset. In case it is observed that there is a difference in the effectiveness of the technique when applied to different datasets, it would be also interesting to analyze the reason for that observation.
- In this research, the Heap Maps are generated for political signals. It would be interesting to consider other forms of signals such as social or economic signals and study the effectiveness of the proposed technique for these other types of signals.

This dissertation further studies two problems deeper in the domain of influence propagation. Given a social network and a two player setting let us consider that the first player has already selected her seed set of nodes. The first problem considers that the goal of the second player is to contain the *reach* of the first player within the social network community. This problem is also relevant for containment of disease in epidemiology, containment of forest fire and several other domains. The second problem is to selectively incentivize a set of nodes for the second player such that under the Separated Threshold model of propagation and with minimum budget, the second player is able to garner more loyal followers than the first player. These works consider that the first player is oblivious of the second player. Future work in this direction could be the following:

- Consider that the first player is aware that the second player will be coming next and that the second player's objective is to get a larger market share compared to the first player. Consider that the first player has a budget $B$. Then, how should the first player select her seed set of $B$ nodes such that the first player is able to maximize the minimum investment (size of seed set) that the second player needs to make in order to get a larger share of the market compared to the first player?
- Consider that the first player is aware that the second player will be coming next and that the second player has a budget $B$. With this knowledge, how should the first player select the smallest sized seed set such that at the end of the observation period, the first player is able to capture a bigger market share compared to the second player?

REFERENCES


[1] S. Banerjee, S. Shirazipourazad, and A. Sen, "On region-based fault tolerant design of distributed file storage in networks," in *INFOCOM, 2012 Proceedings IEEE*, IEEE, 2012, pp. 2806–2810.

[2] D. Patterson, G. Gibson, and R. Katz, "A case for redundant arrays of inexpensive disks (RAID)," in *Proceedings of ACM SIGMOD International conference on Management of data*, 1988, pp. 109–116.

[3] Q. Malluhi and W. Johnston, "Coding for high availability of a distributed-parallel storage system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 12, pp. 1237–1252, 1998.

[4] A. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized erasure codes for distributed networked storage," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, no. SI, pp. 2809–2816, 2006.

[5] A. Jiang and J. Bruck, "Diversity coloring for distributed storage in mobile networks," Tech. Rep., 2001.

[6] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 476–489, 2011.

[7] S. Pawar, S. El Rouayheb, and K. Ramchandran, "On secure distributed data storage under repair dynamics," in *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, 2010, pp. 2543–2547.

[8] A. Sen, B. Shen, L. Zhou, and B. Hao, "Fault-tolerance in sensor networks: A new evaluation metric," in *Proceedings of IEEE INFOCOM*, 2006.

[9] M. Naor and R. Roth, "Optimal File Sharing in Distributed Networks," *SIAM Journal on Computing*, vol. 24, pp. 158–183, 1995.

[10] A. Jiang and J. Bruck, "Network file storage with graceful performance degradation," *ACM Transactions on Storage (TOS)*, vol. 1, no. 2, pp. 171–189, 2005.

[11] M. Sardari, R. Restrepo, F. Fekri, and E. Soljanin, "Memory allocation in distributed storage networks," in *Proceedings IEEE International Symposium on Information Theory (ISIT), 2010*, 2010, pp. 1958–1962.

[12] (). Level 3 Communications, Network Map, [Online]. Available: http://www.level3.com/Resource-Library/Maps/Level-3-Network-Map.aspx.

[13] S. Neumayer and E. Modiano, “Network reliability with geographically correlated failures,” in *INFOCOM, 2010 Proceedings IEEE*, IEEE, 2010, pp. 1–9.

[14] A. Sen, S. Murthy, and S. Banerjee, “Region-based connectivity-a new paradigm for design of fault-tolerant networks,” in *Proceedings of IEEE HPSR, 2009*, 2009, pp. 1–7.

[15] P. Agarwal, A. Efrat, S. K. Ganjugunte, D. Hay, S. Sankararaman, and G. Zussman, “The resilience of WDM networks to probabilistic geographical failures,” *IEEE/ACM Transactions on Networking*, vol. 21, no. 5, pp. 1525–1538, 2013.

[16] S. Neumayer, G. Zussman, R. Cohen, and E. Modiano, “Assessing the vulnerability of the fiber infrastructure to disasters,” *IEEE/ACM Transactions on Networking*, vol. 19, no. 6, pp. 1610–1623, 2011.

[17] E. K. Cetinkaya, D. Broyles, A. Dandekar, S. Srinivasan, and J. P. Sterbenz, “A comprehensive framework to simulate network attacks and challenges,” in *Reliable Networks Design and Modeling (RNDM)*, 2010, pp. 538–544.

[18] D. Zhang, S. A. Gogi, D. S. Broyles, E. K. Cetinkaya, and J. P. Sterbenz, “Modelling attacks and challenges to wireless networks,” in *4th Reliable Networks Design and Modeling (RNDM)*, 2012, pp. 806–812.

[19] Y. Cheng, M. T. Gardner, J. Li, R. May, D. Medhi, and J. P. Sterbenz, “Optimised heuristics for a geodiverse routing protocol,” in *10th International Conference on the Design of Reliable Communication Networks (DRCN)*, 2014, pp. 1–9.

[20] S. El Rouayheb and K. Ramchandran, “Fractional repetition codes for repair in distributed storage systems,” in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, 2010.

[21] A. Jiang, M. Cook, and J. Bruck, “Optimal t-interleaving on tori,” in *Proceedings. International Symposium on Information Theory, 2004. ISIT 2004.*, 2004, p. 22.

[22] A. Jiang and J. Bruck, “Memory allocation in information storage networks,” in *Proceedings. IEEE International Symposium on Information Theory, 2003.*, 2003, p. 453.

[23] S. Banerjee, S. Shirazipourazad, P. Ghosh, and A. Sen, “Beyond connectivity - new metrics to evaluate robustness of networks,” in *Proceedings of IEEE HPSR, 2011*, 2011.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 1990.

[25] M. Garey and D. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Company, 1979.

[26] A. Haque, P.-H. Ho, and H. M. Alazemi, "Inter group shared protection (I-GSP) for survivable WDM mesh networks," *Optical Switching and Networking*, vol. 10, no. 2, pp. 119–131, 2013.

[27] E. L. Lloyd and G. Xue, "Relay node placement in wireless sensor networks," *Computers, IEEE Transactions on*, vol. 56, no. 1, pp. 134–138, 2007.

[28] S. Khuller and B. Raghavachari, "Improved approximation algorithms for uniform connectivity problems," in *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, ACM, 1995, pp. 1–10.

[29] W. Zhang, G. Xue, and S. Misra, "Fault-tolerant relay node placement in wireless sensor networks: Problems and algorithms," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, IEEE, 2007, pp. 1649–1657.

[30] X. Han, X. Cao, E. L. Lloyd, and C.-C. Shen, "Fault-tolerant relay node placement in heterogeneous wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 9, no. 5, pp. 643–656, 2010.

[31] S. Misra, S. D. Hong, G. Xue, and J. Tang, "Constrained relay node placement in wireless sensor networks: Formulation and approximations," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 2, pp. 434–447, 2010.

[32] J.-Y. Chang and Y.-W. Chen, "A cluster-based relay station deployment scheme for multi-hop relay networks," *Communications and Networks, Journal of*, vol. 17, no. 1, pp. 84–92, 2015.

[33] P. Li, C. Huang, and Q. Liu, "Bcdp: Budget constrained and delay-bounded placement for hybrid roadside units in vehicular ad hoc networks," *Sensors*, vol. 14, no. 12, pp. 22 564–22 594, 2014.

[34] G.-H. Lin and G. Xue, "Steiner tree problem with minimum number of steiner points and bounded edge-length," *Information Processing Letters*, vol. 69, no. 2, pp. 53–57, 1999.

[35] D. Chen, D.-Z. Du, X.-D. Hu, G.-H. Lin, L. Wang, and G. Xue, "Approximations for steiner trees with minimum number of steiner points," *Journal of Global Optimization*, vol. 18, no. 1, pp. 17–33, 2000.

[36] J. S. Mitchell, "Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric tsp, k-mst, and related problems," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1298–1309, 1999.

[37] S. Arora, "Polynomial time approximation schemes for euclidean tsp and other geometric problems," in *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, IEEE, 1996, pp. 2–11.

[38] F. A. Chudak, T. Roughgarden, and D. P. Williamson, "Approximate k-msts and k-steiner trees via the primal-dual method and lagrangean relaxation," *Mathematical Programming*, vol. 100, no. 2, pp. 411–421, 2004.

[39] 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.

[40] J. Gao, S. Buldyrev, H. Stanley, and S. Havlin, "Networks formed from interdependent networks," *Nature Physics*, vol. 8, no. 1, pp. 40–48, 2011.

[41] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. Porcellinis, and R. Setola, "Modelling interdependent infrastructures using interacting dynamical models," *International Journal of Critical Infrastructures*, vol. 4, no. 1-2, pp. 63–79, 2008.

[42] M. Parandehgheibi and E. Modiano, "Robustness of interdependent networks: The case of communication networks and the power grid," Tech. Rep., 2013, pp. 2164–2169.

[43] D. Nguyen, Y. Shen, and M. Thai, *Detecting critical nodes in interdependent power networks for vulnerability assessment*. 2013.

[44] J.-F. Castet and J. H. Saleh, "Interdependent multi-layer networks: Modeling and survivability analysis with applications to space-based networks," *PloS one*, vol. 8, no. 4, e60402, 2013.

[45] A. Sen, A. Mazumder, J. Banerjee, A. Das, and R. Compton, "Identification of k most vulnerable nodes in multi-layered network using a new model of interdependency," in *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, IEEE, 2014, pp. 831–836.

[46] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed nets," *Journal of theoretical biology*, vol. 22, no. 3, pp. 437–467, 1969.

[47] S. R., "U. s. risks national blackout from small-scale attack," *Wall Street Journal*, vol. 02304020104579433670284061220, 2012. [Online]. Available: http://online.wsj.com/news/articles/SB100014240527.

[48] A. Bernstein, D. Bienstock, D. Hay, M. Uzunoglu, and G. Zussman, "Power grid vulnerability to 3 geographically correlated failures-analysis and control implications.," arXiv:1206.1099, preprint, 2012.

[49] U. Feige, L. Lovász, and P. Tetali, "Approximating min sum set cover," *Algorithmica*, vol. 40, no. 4, pp. 219–234, 2004.

[50] R. Hassin and A. Levin, "An approximation algorithm for the minimum latency set cover problem," in *Algorithms-ESA*, Berlin: Springer, 2005, pp. 726–733.

[51] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing, "A latent variable model for geographic lexical variation," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2010, pp. 1277–1287.

[52] Z. Cheng, J. Caverlee, and K. Lee, "You are where you tweet: A content-based approach to geo-locating twitter users," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, ACM, 2010, pp. 759–768.

[53] D. Yang, D. Zhang, Z. Yu, and Z. Wang, "A sentiment-enhanced personalized location recommendation system," in *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, ACM, 2013, pp. 119–128.

[54] Y. Li, M. Steiner, L. Wang, Z. L. Zhang, and J. Bao, "Dissecting foursquare venue popularity via random region sampling," in *Proceedings of the 2012 ACM conference on CoNEXT student workshop*, ACM, 2012, pp. 21–22.

[55] R. Priedhorsky, A. Culotta, and S. Y. Del Valle, "Inferring the origin locations of tweets with quantitative confidence," arXiv preprint arXiv:1305.3932, preprint, 2013.

[56] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis, "Discovering geographical topics in the twitter stream," in *Proceedings of the 21st international conference on World Wide Web*, ACM, 2012, pp. 769–778.

[57] Z. Yin, L. Cao, J. Han, C. Zhai, and T. Huang, "Geographical topic discovery and comparison," in *Proceedings of the 20th international conference on World Wide Web*, ACM, 2011, pp. 247–256.

[58] M. C. Gonzalez, C. A. Hidalgo, and A. L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008.

[59] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: User movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2011, pp. 1082–1090.

[60] A. Sadilek, H. Kautz, and J. P. Bigham, "Finding your friends and following them to where you are," in *Proceedings of the fifth ACM international conference on Web search and data mining*, ACM, 2012, pp. 723–732.

[61] A. Noulas, S. Scellato, R. Lambiotte, M. Pontil, and C. Mascolo, "A tale of many cities: Universal patterns in human urban mobility," *PloS one*, vol. 7, no. 5, e37027, 2012.

[62] H.-w. Chang, D. Lee, M. Eltaher, and J. Lee, "@ phillies tweeting from philly? predicting twitter user locations with spatial word usage," in *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, IEEE Computer Society, 2012, pp. 111–118.

[63] J. Mahmud, J. Nichols, and C. Drews, "Where is this tweet from? inferring home locations of twitter users.," *ICWSM*, vol. 12, pp. 511–514, 2012.

[64] A. Boutet, H. Kim, and E. Yoneki, "What's in twitter, i know what parties are popular and who you are supporting now!," 4, vol. 3, Springer, 2013, pp. 1379–1391.

[65] J. Brynielsson, A. Horndahl, F. Johansson, L. Kaati, C. Mårtenson, and P. Svenson, "Analysis of weak signals for detecting lone wolf terrorists," pp. 197–204, 2012.

[66] J. Dahlin, F. Johansson, L. Kaati, C. Martenson, and P. Svenson, "Combining entity matching techniques for detecting extremist behavior on discussion boards," in *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, IEEE Computer Society, 2012, pp. 850–857.

[67] J. M. Xu, A. Bhargava, R. Nowak, and X. Zhu, "Socioscope: Spatio-temporal signal recovery from social media," in, M. Learning and K. D. in Databases . Springer Berlin Heidelberg, Eds., 2012, pp. 644–659.

[68] M. Thelwall, K. Buckley, G. Paltoglou, M. Skowron, D. Garcia, S. Gobron, J. Ahn, A. Kappas, D. Küster, and J. A. Holyst, "Damping sentiment analysis in online communication: Discussions, monologs and dialogs," in *International Conference on Intelligent Text Processing and Computational Linguistics*, 2013, pp. 1–12.

[69] P. Melville, W. Gryc, and R. D. Lawrence, "Sentiment analysis of blogs by combining lexical knowledge with text classification," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2009, pp. 1275–1284.

[70] M. Joshi, D. Das, K. Gimpel, and N. A. Smith, "Movie reviews and revenues: An experiment in text regression," pp. 293–296, 2010.

[71] S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. A. Smith, "Predicting risk from financial reports with regression," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2009, pp. 272–280.

[72] T. Zhang, "Some sharp performance bounds for least squares regression with l1 regularization," *The Annals of Statistics*, vol. 37, no. 5, pp. 2109–2144, 2009.

[73] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale-regularized least squares," *IEEE journal of selected topics in signal processing*, vol. 1, no. 4, pp. 606–617, 2007.

[74] J. Z. Kolter and A. Y. Ng, "Regularization and feature selection in least-squares temporal difference learning," in *Proceedings of the 26th annual international conference on machine learning*, ACM, 2009, pp. 521–528.

[75] J. R. Brzezinski and G. J. Knafl, "Logistic regression modeling for context-based classification," pp. 755–759, 1999.

[76] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of statistical software*, vol. 33, no. 1, p. 1, 2010.

[77] J. Liu, S. Ji, J. Ye, *et al.*, *Slep: Sparse learning with efficient projections*. 2009, vol. 6, p. 491.

[78] 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.

[79] T. Carnes, C. Nagarajan, S. M. Wild, and A. Van Zuylen, "Maximizing influence in a competitive social network: A follower's perspective," in *Proceedings of the ninth international conference on Electronic commerce*, ACM, 2007, pp. 351–360.

[80] X. He, G. Song, W. Chen, and Q. Jiang, "Influence blocking maximization in social networks under the competitive linear threshold model.," in *SDM*, 2012, pp. 463–474.

[81] G. Tuli, C. J. Kuhlman, M. V. Marathe, S. Ravi, and D. J. Rosenkrantz, "Blocking complex contagions using community structure," 2012.

[82] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proceedings of the 19th international conference on World wide web*, ACM, 2010, pp. 631–640.

[83] M. Kimura, K. Saito, and R. Nakano, "Extracting influential nodes for information diffusion on a social network," in *AAAI*, vol. 7, 2007, pp. 1371–1376.

[84] P. Shakarian and D. Paulo, "Large social networks can be targeted for viral marketing with small seed sets," in *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, IEEE Computer Society, 2012, pp. 1–8.

[85] S. Shirazipourazad, B. Bogard, H. Vachhani, A. Sen, and P. Horn, "Influence propagation in adversarial setting: How to defeat competition with least amount of investment," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, ACM, 2012, pp. 585–594.

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

[87] M. Jain, D. Korzhyk, O. Vaněk, V. Conitzer, M. Pěchouček, and M. Tambe, "A double oracle algorithm for zero-sum security games on graphs," in *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, International Foundation for Autonomous Agents and Multiagent Systems, 2011, pp. 327–334.

[88] S. Wang, X. Zhao, Y. Chen, Z. Li, K. Zhang, and J. Xia, "Negative influence minimizing by blocking nodes in social networks," in *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[89] A. Hayrapetyan, D. Kempe, M. Pál, and Z. Svitkina, "Unbalanced graph cuts," in *Algorithms–ESA 2005*, Springer, 2005, pp. 191–202.

[90] M. Conforti, M. Di Summa, F. Eisenbrand, and L. A. Wolsey, "Network formulations of mixed-integer programs," *Mathematics of Operations Research*, vol. 34, no. 1, pp. 194–209, 2009.

[91] M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM review*, vol. 33, no. 1, pp. 60–100, 1991.

[92] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[93] P. Erd6s and A. Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hungar. Acad. Sci*, vol. 5, pp. 17–61, 1960.

[94] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[95] S. Bharathi, D. Kempe, and S. Mahyar, *Competitive influence maximization in social networks*. 2007.

[96] T. Carnes, C. Nagarajan, S. M. Wild, and A. van Zuylen, "Maximizing influence in a competitive social network: A follower's perspective," in *Proceedings of the ninth international conference on Electronic commerce*, ser. ICEC '07, 2007, pp. 351–360.

[97] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09, 2009, pp. 199–208.

[98] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '01, 2001, pp. 57–66.

[99] A. Goyal, F. Bonchi, L. V. Lakshmanan, and S. Venkatasubramanian, "Approximation analysis of influence spread in social networks," *ArXiv preprint arXiv:1008.2005*, 2010.

[100] D. Kempe, J. Kleinberg, and E. 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*, ser. KDD '03, 2003, pp. 137–146.

[101] M. Mathioudakis, F. Bonchi, C. Castillo, A. Gionis, and A. Ukkonen, "Sparsification of influence networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '11, 2011, pp. 529–537.

[102] A. Borodin, Y. Filmus, and J. Oren, "Threshold models for competitive influence in social networks," in *Internet and network economics*, Springer, 2010, pp. 539–550.

[103] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2007, pp. 420–429.

[104] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2010, pp. 1029–1038.

[105] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, IEEE, 2010, pp. 88–97.

[106] A. Goyal, W. Lu, and L. V. Lakshmanan, "Simpath: An efficient algorithm for influence maximization under the linear threshold model," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, IEEE, 2011, pp. 211–220.

[107] S. Bhagat, A. Goyal, and L. V. Lakshmanan, "Maximizing product adoption in social networks," in *Proceedings of the fifth ACM international conference on Web search and data mining*, ser. WSDM '12, 2012, pp. 603–612.

[108] A. Banerjee, A. G. Chandrasekhar, E. Duflo, and M. O. Jackson, "The diffusion of microfinance," *Science*, vol. 341, no. 6144, 2013.

[109] P. Zhang, W. Chen, X. Sun, Y. Wang, and J. Zhang, "Minimizing seed set selection with probabilistic coverage guarantee in a social network," in *Proceedings of the ACM SIGKDD*, ACM, 2014.

[110] S. Aral and D. Walker, "Identifying influential and susceptible members of social networks," *Science*, vol. 337, no. 6092, pp. 337–341, 2012.

[111] H. Habiba, Y. Yu, T. Y. Berger-Wolf, and J. Saia, "Finding spread blockers in dynamic networks," in *Proceedings of the Second international conference on Advances in social network mining and analysis*, ser. SNAKDD'08, 2010, pp. 55–76.

[112] M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical review E*, vol. 74, no. 3, p. 036 104, 2006.

[113] ——, “The structure of scientific collaboration networks,” *Proceedings of the National Academy of Sciences*, vol. 98, no. 2, pp. 404–409, 2001.