Identifying Sources of Anomalies in Complex Networks

by

Kaustav Basu

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

Approved June 2022 by the
Graduate Supervisory Committee:

Arunabha Sen, Chair
Hasan Davulcu
Huan Liu
Guoliang Xue

ARIZONA STATE UNIVERSITY

August 2022

©2022 Kaustav Basu

All Rights Reserved

ABSTRACT

The problem of monitoring complex networks for the detection of anomalous behavior is well known. Sensors are usually deployed for the purpose of monitoring these networks for anomalies and Sensor Placement Optimization (SPO) is the problem of determining where these sensors should be placed (deployed) in the network. Prior works have utilized the well known Set Cover formulation in order to determine the locations where sensors should be placed in the network, so that anomalies can be effectively detected. However, such works cannot be utilized to address the problem when the objective is to not only detect the presence of anomalies, but also to detect (distinguish) the source(s) of the detected anomalies, i.e., uniquely monitoring the network.

In this dissertation, I attempt to fill in this gap by utilizing the mathematical concept of Identifying Codes and illustrating how it not only can overcome the aforementioned limitation, but also it, and its variants, can be utilized to monitor complex networks modeled from multiple domains. Over the course of this dissertation, I make key contributions which further enhance the efficacy and applicability of Identifying Codes as a monitoring strategy. First, I show how Identifying Codes are superior to not only the Set Cover formulation but also standard graph centrality metrics, for the purpose of uniquely monitoring complex networks. Second, I study novel problems such as the budget constrained Identifying Code, scalable Identifying Code, robust Identifying Code etc., and present algorithms and results for the respective problems. Third, I present useful Identifying Code results for restricted graph classes such as Unit Interval Bigraphs and Unit Disc Bigraphs. Finally, I show the universality of Identifying Codes by applying it to multiple domains.

DEDICATION

To my late grandfathers, who would have enjoyed reading this thesis the most.

ACKNOWLEDGMENTS

I am sincerely grateful to my advisor, Prof. Arunabha Sen, for giving me the opportunity to work with him. His thought clarity, knack of dissecting complex real world problems into simple mathematical abstractions and patience, are some of the many attributes of his, which has influenced me greatly over the years. I will always cherish our countless hours of fruitful discussions (research and non research based) and looking back, I can clearly see how working closely with Dr. Sen has not only enhanced my research capabilities, but also made me an overall better person. I am truly honored to have Profs. Hasan Davulcu, Huan Liu and Guoliang Xue as my committee members. I am thankful to them for their insights, feedback and their overall support towards my research. Every interaction with them, be it in class or during meetings, has been enriching and always left me with means to further enhance the problems studied in this dissertation. Finally, to Monica, Pamela, Jaya and Arzuhan, for every way in which you helped me over the past 6 years, thank you.

I had extremely helpful (ex) senior PhD students, none more than Dr. Arindam Mitra, Dr. Sailik Sengupta and Mr. Sandipan Choudhuri, who have continuously supported, pushed and critiqued me in equal measures, throughout my arduous PhD journey. Drs. Anisha Mazumder, Arun Das, Joydeep Banerjee and Chenyang Zhou were my initial collaborators and working with them was a memorable experience, albeit included a steep learning curve. Sohini, Suli and Ali all made our weekly research meetings insightful and I look forward to collaborating with them in the future. Personally, I was truly fortunate to have an amazing set of friends. Anket, Anoop and Vinay were the first friends I had in Tempe and they were kind, patient, made me feel at home and played a major hand in moulding me to be the independent person I am today. Parijat, Jayeeta, Debatrayee, Srijita, Sanghita and Somnath, made me realize that even though I was far away from home, there was still something from home to look forward to in Tempe.

Sumon Chaudhuri, Proyag Pal and Arjunil Pathak have been my brothers-in-arms and my support system for more than a decade now. They have celebrated with me at my best, and they have commiserated with me during my worst. Their unwavering support and friendship was even more appreciated during the pandemic years, when we came together virtually every weekend, to talk about everything and also nothing. No amount of thanks will ever be sufficient for these three brilliant, wonderful, kind hearted and sometimes irritating brothers.

Initially, my parents did not want me to pursue higher studies on the other side of the planet. They were naturally worried but nonetheless, they acknowledged and supported my choice. I cannot begin to truly comprehend their sacrifice which allowed me to pursue my own dreams in a distant foreign land. I could see age catching up with them, whenever I saw them on a yearly basis, and all of its related hardships, which made me question my choice to leave them behind on numerous occasions. Whenever I voiced these concerns to them, they countered with the fact that nothing would make them happier to see me succeed in life. As I wrap up my PhD career, I only hope that I have managed to live up to their expectations and made them proud. I could not have asked for a more understanding, loving, supporting and sometimes irrational set of parents. Ma and Baba, thank you. To my extended family, I love you all and thank you for everything that you have done for me over the years.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

Complex interactive systems are often modeled as networks (or graphs), such as the critical infrastructure networks, human-human interaction networks, online social networks, etc. These "complex networks" are ubiquitous and often require efficient strategies to monitor its operational behavior. For instance, law enforcement agencies monitor networks of suspect individuals in order to prevent any terror/drug related activities, power utility companies monitor their network (or grid) continuously in order to quickly detect any kind of fault(s), online social networking platforms religiously monitor user content to quickly detect fake news/cyber bullying/hate speech propagation, etc. Depending on the domain from which the complex network has been modeled from, the term "anomalous behavior" can refer to - (i) suspect individuals becoming active in plotting terror attacks or dealing and distributing drugs, in the case of human-human interaction networks, (ii) propagation of the fault signals from power grid entities (substations, buses, transformers, etc.) in critical infrastructure networks, (iii) propagation of fake news/cyber bullying/hate speech, etc. from users on social networking platforms. If allowed to manifest for a certain period of time in the complex network, the anomaly may cause large scale disruption and harm to society in general. Therefore, it is a fundamental undertaking to not only detect the manifestations of anomalous behavior(s) in such networks, but also to *identify its source(s)*, in order to correct the sequence of events that started this anomaly, i.e., take preventive measures. Over the course of this thesis, we show how sensors can be optimally deployed in order to monitor a particular complex network, such that, if any node or entity in the network were to behave anomalously, then the deployed sensors can *uniquely identify which node (entity) triggered the anomalous behavior*. Note that, certain assumptions have been made in most of the problems addressed in this thesis. Firstly, only one node (or entity) can become active in anomalous behavior at a given instance of time. Secondly, the signal or "cues" of the activation of a node, is propagated to all of its neighbors, reasons for which will be detailed for each domain under study.

1.1 Motivation

Certain real world incidences motivated the work which resulted in the chapters presented as a part of this thesis. We refer to two incidences - (i) a transformer failure on Salt River Project's (SRP) network, and (ii) contamination of water distribution networks in the United States, Canada and Finland.

1.1.1 SRP Transformer Failure

In the early hours of June 1, 2016, a large power transformer at the Rudd substation of Salt River Project (SRP), a large utility company in Arizona, suddenly caught fire (News 2016). A 27,000-gallon tank of mineral oil used as a transformer coolant, burned and spewed thick smoke over a large area. A few snapshots are illustrated in Figure 1. The cause of the failure was identified to be bushing failure. Due to the redundancy present in the system design, as well as the fact that the fire broke out during low-load conditions (system load is small in early morning), no power outages

Figure 1. Transformer fire at Salt River Project's Rudd substation in Avondale, AZ

occurred. This incident highlights the need for better monitoring techniques for the critical entities in the network.

SRP shared their operational data leading up to the failure of this transformer for analysis. A snapshot of the operational data is presented in Figure 2. This snapshot is a visualization of the variation in the Signal to Noise Ratio (SNR) and it can be observed that the width of the ratio increased as the transformer headed to failure. Since causes of such failures gradually build-up over time, the signs of an impending failure may be observable "days" before the actual failure event. Phasor Measurement Units (or PMUs in short), are sensors deployed on entities in the power grid and they continuously produce outputs at a very fast rate (typically 30 samples per second). When placed near transformers, PMUs, through their measurements, can serve as *sensors* to monitor the behavior of the transformer, and capture the anomalous signals of anomalies before it causes large scale failure(s).

1.1.2 Contamination of Water Distribution Networks

Accidental or incidental contamination of water distribution networks may cause severe harm to it's direct consumers. For instance, the spread of contaminated

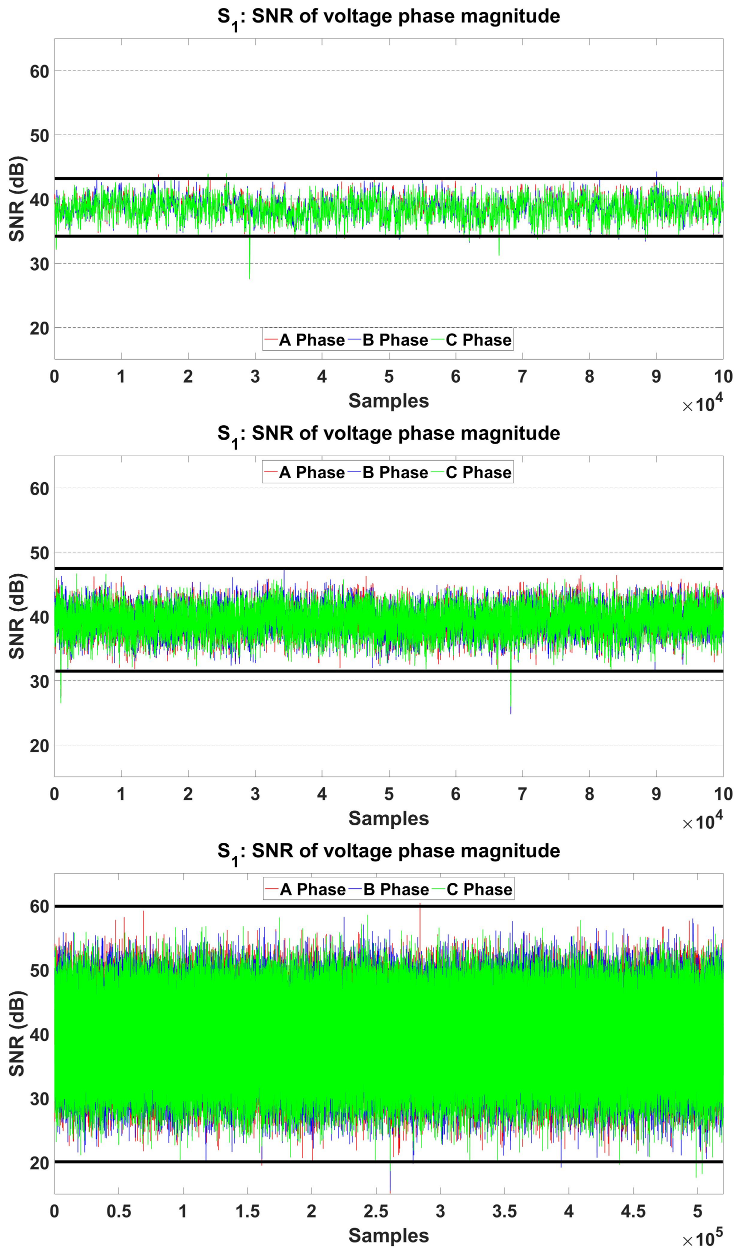


Figure 2. Variation in width of SNR of SRP Data.

water affected more than 400,000 residents in Milwaukee in 1993. This was caused by a microorganism which was transported through the distribution system. In another incident, contaminated drinking water affected over 2000 people in Walkerton, Ontario, Canada. Moreover, in 2007, 8,500 people were ill in Nokia, Finland due to a cross-connection of wastewater into the distribution network (Kauppinen et al. 2019). Sensors are usually placed in water distribution networks to not only monitor the flow of water, but also to monitor the presence of contaminants and leaks in the system. Such anomalies in the water distribution network must quickly be detected and its origin accurately identified in order to mitigate the spread of contaminants or leaks.

Since complex networks can be modeled as a graph, unique monitoring of nodes in the network can be accomplished by deploying sensors on all the nodes of the network, provided the graph admits an Identifying Code (the necessary and sufficient condition for the same are discussed later). However, each sensor has a cost associated with it and as a result, it is important to determine the minimum number of entities where sensors can be placed. Further, given that the anomalous behavior of an entity can be observed by sensors located within a certain distance of the entity in question, the propagation of signals (indicating the anomalous behavior), can be utilized to deploy effective monitoring strategies, so that an alarm is generated before an entity reaches a critical state. Identifying Code is a mathematical tool that can be used for monitoring entities in complex network systems. Using this technique, the fewest number of sensors needed, to enable an operator to uniquely identify the abnormal entity, can be determined and deployed in the network.

1.2 Contributions

The contributions of this dissertation can be summarized below:

- *Human-Human Interaction Network Research*: In this domain, we show how the notion of Identifying Codes can be effectively utilized for monitoring illicit networks such as Drug Trafficking and Terror networks. Here, the sensors are law enforcement agents who have to be deployed to monitor certain suspect individuals. Additionally, we provide a lower bound for the robust Identifying Code problem, specifically targeted for the scenario where the sensors can fail. Next, we address the problem where we allow multiple simultaneous nodes to become active in anomalous behavior. Finally, we present a more realistic version of the Identifying Code problem, called the Augmented Identifying Code problem and show how our approaches are superior to standard graph centrality metrics, which have been used in prior studies for monitoring suspect individuals.
- *Critical Infrastructure Network Research*: A variant of Identifying Code, called Discriminating Code, is utilized for the problems in this domain. This variant allows the monitoring of only a subset of the nodes in the network, as opposed to all the nodes in the network. We show how this approach can be utilized to uniquely monitor the health of critical power system equipments (High Voltage Transformers, Generating Stations, Transmission Lines, etc.). Moreover, we study an adversarial scenario, where we assume the presence of an attacker, who is trying to prevent the unique monitoring of the critical infrastructure equipments by disabling a sensor.
- *Online Social Network Research*: In this domain, we study the long standing problem of misinformation propagation. In our previous works, we utilized an

Integer Linear Program to determine the optimal number of sensors required for uniquely monitoring the network. However, such approaches fail in this context due to the sheer size of the online social network. As a result, novel approximation algorithms and heuristics were designed and presented. We showed that our heuristics provided results which were comparable to the approximation algorithm, all the while taking a fraction of the time taken by the approximation algorithm.

- *Water Distribution Network Research*: Here, we address the problem of monitoring the water distribution network for contamination and leaks. Our study was inspired from the Battle of the Water Sensor Networks challenge (Ostfeld et al. 2008). We present a novel budget constraint version of the Identifying Code problem, where the operator of the complex network cannot place more than k sensors in the network. The goal here is to maximize the number of nodes being uniquely monitored, as uniquely monitoring the entire network is not possible. Moreover, we show that a constant factor approximation algorithm is not possible for the problem unless P = NP.
- *Identifying Code Problems for Restricted Graph Classes*: We study the problem of monitoring points of interest in one and two dimensions. We show how this can be modeled by Unit Interval Bigraphs and Unit Disc Bigraphs respectively. For the former, we provide a novel polynomial time optimal algorithm utilizing Dynamic Programming. For the latter we provide an Integer Linear Program for the optimal computation for the minimum number of sensors required to uniquely monitor the points of interest. Additionally, we study the problem of monitoring the surface of the earth, via satellites, in this section. We model the

earth using a soccer ball and present various results, such as upper and lower bounds for the graph created from this soccer ball.

Several studies from the aforementioned research resulted in publications in peer reviewed workshops/conferences/journals and are listed below:

- *Human-Human Interaction Network Research*:
  1. Arunabha Sen et al. 2018. "Terrorist Network Monitoring with Identifying Code." In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation,* 329–339. Springer
  2. Kaustav Basu, Chenyang Zhou, et al. 2018. "A novel graph analytic approach to monitor terrorist networks." In *2018 IEEE International Conference on Social Computing & Networking (SocialCom),* 1159–1166. IEEE
  3. Kaustav Basu and Arunabha Sen. 2019a. "Monitoring individuals in drug trafficking organizations: a social network analysis." In *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM),* 480–483. IEEE
  4. Kaustav Basu and Arunabha Sen. 2019b. "On augmented identifying codes for monitoring drug trafficking organizations." In *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM),* 1136–1139. IEEE
  5. Kaustav Basu and Arunabha Sen. 2021b. "Identifying individuals associated with organized criminal networks: a social network analysis." *Social Networks* 64:42–54
- *Critical Infrastructure Network Research*:

1. Kaustav Basu, Malhar Padhee, et al. 2018. “Health Monitoring of Critical Power System Equipments Using Identifying Codes.” In *International Conference on Critical Information Infrastructures Security,* 29–41. Springer
2. Sailik Sengupta, Kaustav Basu, et al. 2020. “Moving target defense for robust monitoring of electric grid transformers in adversarial environments.” In *International Conference on Decision and Game Theory for Security,* 241–253. Springer

- *Online Social Network Research*:

1. Kaustav Basu. 2019. “Identification of the Source (s) of Misinformation Propagation Utilizing Identifying Codes.” In *Companion Proceedings of The 2019 World Wide Web Conference,* 7–11
2. Kaustav Basu and Arunabha Sen. 2021a. “Epidemiological Model Independent Misinformation Source Identification”

- *Water Distribution Network Research*:

1. Kaustav Basu and Arunabha Sen. 2022. “Sensor Network Design for Uniquely Identifying Sources of Contamination in Water Distribution Networks”

- *Identifying Code Problems for Restricted Graph Classes*:

1. Kaustav Basu et al. 2019. “Sensor Networks for Structural Health Monitoring of Critical Infrastructures Using Identifying Codes.” In *2019 15th International Conference on the Design of Reliable Communication Networks (DRCN),* 43–50. IEEE
2. Arunabha Sen et al. 2019. “On upper and lower bounds of identifying code set for soccer ball graph with application to satellite deployment.” In

Proceedings of the 20th International Conference on Distributed Computing and Networking, 307–316. ACM

I was fortunate to have collaborated on other studies during my time as PhD student at ASU. Some of them are presented below:

1. Joydeep Banerjee et al. 2017. "Finding K Contingency List in Power Networks using a New Model of Dependency." *arXiv preprint arXiv:1705.07410*
2. Joydeep Banerjee, Kaustav Basu, and Arunabha Sen. 2018a. "Analysing robustness in intra-dependent and inter-dependent networks using a new model of interdependency." *International Journal of Critical Infrastructures* 14 (2): 156–181
3. Joydeep Banerjee, Kaustav Basu, and Arunabha Sen. 2018b. "On hardening problems in critical infrastructure systems." *International Journal of Critical Infrastructure Protection* 23:49–67
4. Malhar Padhee et al. 2018. "A new model to analyze power system dependencies." In *2018 IEEE Texas Power and Energy Conference (TPEC),* 1–6. IEEE
5. Chenyang Zhou et al. 2018. "Relay node placement under budget constraint." In *Proceedings of the 19th International Conference on Distributed Computing and Networking,* 1–11
6. Kaustav Basu, Sandipan Choudhuri, et al. 2018. "Insights from statistical analysis of opioid data." *arXiv preprint arXiv:1805.05509*
7. Sandipan Choudhuri et al. 2019. "Predicting Future Opioid Incidences Today." *arXiv preprint arXiv:1906.08891*
8. Arunabha Sen and Kaustav Basu. 2019. "On connectivity of interdependent networks." In *2019 IEEE Global Communications Conference (GLOBECOM),* 1–6. IEEE

9. Arunabha Sen, Sandipan Choudhuri, and Kaustav Basu. 2020. “Structural Dependency Aware Service Chain Mapping for Network Function Virtualization.” In *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020,* 1–6. IEEE
10. Arunaba Sen et al. 2020. “On the Number of Steiner Trees in a Graph.” In *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020,* 1–5. IEEE
11. Malhar Padhee et al. 2020. “Identifying Unique Power System Signatures for Determining Vulnerability of Critical Power System Assets.” *ACM SIGMETRICS Performance Evaluation Review* 47 (4): 8–11
12. A Sen et al. 2021. “Optimal Cost Network Design for Bounded Delay Data Transfer from PMU to Control Center.” In *2021 IEEE Global Communications Conference (GLOBECOM),* 1–6. IEEE

1.3 Dissertation Outline

The chapters presented in this thesis broadly covers four application domain areas – human-human interaction networks, critical infrastructure networks, online social networks and water distribution networks. Apart from these broad domains, we also study the Identifying Code problem for restricted graph classes. More specifically, the organization of the chapters are as follows.

Chapter 2 covers the mathematical formulations of the Identifying Code problem and its variant, the Discriminating Code problem. The necessary and sufficient conditions for a graph to exhibit an Identifying Code is also discussed. Two approaches are presented which have been used in the later chapters to determine the minimum

Identifying Code set of a network (or graph). Finally, the rationale for utilizing Identifying Code, as opposed to other monitoring strategies such as the Set Cover based formulation, is provided.

The analyses on Human-Human Interaction Networks are present in Chapter 3. Here, we not only show how Identifying Codes can be utilized for uniquely monitoring the network of suspect individuals but also provide useful results such as the lower bound of k-fault tolerant Identifying Codes. Moreover, we present algorithms to identify two nodes engaging in simultaneous anomalous behavior. We discuss a novel variant of the Identifying Codes problem, called the Augmented Identifying Codes, and show how all our approaches are superior than standard graph centrality metrics, in the case of uniquely monitoring a given network.

Chapter 4 discusses the problem of uniquely monitoring critical power system equipments. The equipment of interest in our study is the High Voltage Transformer. We show how the power grid system can be modeled as a graph and utilize the Discriminating Code problem, a variant of Identifying Codes, for uniquely monitoring the operational health of the transformers. Additionally, we consider an adversarial scenario, where an attacker tries to disable the sensors deployed in the network so that the defender (the power utility companies) cannot uniquely monitor the network for anomalous behavior.

Misinformation propagation and identifying users engaged in such activities has been an important research topic over the past decade. In Chapter 5, we show how Identifying Codes can be utilized for identifying such users. We base our work on epidemiological models, which have been used in order to deploy sensors in the social network graph. We illustrate how our approach is independent of underlying models (such as SI, SIR, SIRS, etc.) and present a novel transformation of the Identifying

Code problem to the Hitting Set problem. Finally, we present a scalable heuristic which scales better than the approximation algorithm, in order to handle the larger graph instances in the social network domain.

Chapter 6 studies identifying locations of contamination and leaks in water distribution networks. Here, we present a novel budget constrained version of the Identifying Code problem, show how this problem is a generalization of the Max k-Cover problem and that no constant factor approximation algorithm can exist for this problem, unless P = NP.

We study the Identifying Code problem for restricted graph classes (Unit Interval Bigraph, Unit Disc Bigraph and Soccer Ball Graph) in Chapter 7, with regards to monitoring points/locations of interest. We present optimal solution techniques and discuss the upper and lower bounds of the minimum Identifying Code set in the case of the soccer ball graph.

Chapter 8 concludes this thesis and discusses the overall objectives of the problems addressed in it. Moreover, we highlight certain problems which can be examined and further pursed.

The following table highlights the numerous notations utilized in this dissertation, for the ease of understanding.

$G = (V, E)$	A graph with the vertex set as V and edge set as E
$N(v)$	Neighborhood set of node v
$N^+[v]$	Closed neighborhood set of node v, i.e., $N(v) \cup v$
$N^{out}(v)$	Out-neighborhood set of node v
$N^{out}[v]$	Closed out-neighborhood set of node v, i.e., $N^{out}(v) \cup v$
$\bigoplus$	Symmetric difference operator
μ	Mean or expected value
σ	Standard deviation
$N^k(v)$	k-hop neighbors of node v
$\mathcal{D}$	Defender
$\mathcal{A}$	Attacker
$A^{\mathcal{D}}$	Pure strategy or the defender's set of actions
$A^{\mathcal{A}}$	Pure strategy or the attacker's set of actions
$R^{\mathcal{D}}$	Defender's utility or rewards
$R^{\mathcal{A}}$	Attacker's utility or rewards
G_{udg}	A unit disc graph
G_{cbp}	A complete bipartite graph
G_{ig}	An interval graph
$P_{i,j}$	j-th pentagonal node on Layer i
$H_{i,j}$	j-th hexagonal node on Layer i

Table 1. Table of Notations

Chapter 2

BACKGROUND

This chapter aims to present to the reader the notion of Identifying Codes and how it can be utilized to uniquely identify anomalies in a complex network. The mathematical concepts of Identifying Codes and its variant, Discriminating Codes, are defined and the necessary and sufficient conditions for the existence of both, are discussed. Moreover, two novel approaches are presented, which have been utilized in later chapters, to determine the Minimum Identifying Code Set (MICS), and the Minimum Discriminating Code Set (MDCS), for a given graph, which admitted an Identifying Code (or a Discriminating Code). Finally, to conclude this chapter, the necessity of Identifying Codes for uniquely monitoring a complex network is presented.

2.1 Preliminaries

A majority of the research done on Identifying Codes is analytical in nature (Karpovsky, Chakrabarty, and Levitin 1998; Laifenfeld and Trachtenberg 2008; Irene Charon, Hudry, and Lobstein 2002; Irène Charon, Hudry, and Lobstein 2003). Sensor placement optimization for the unique identification of the nodes in a graph was first introduced as Identifying Codes in (Karpovsky, Chakrabarty, and Levitin 1998), and provided results for special types of graphs, such as binary cube graphs and trees. The NP Completeness for the MICS problem was proved in (Irène Charon, Hudry, and Lobstein 2003), following a reduction from the 3-SAT problem. Polynomial time approximation algorithms were provided in (Laifenfeld and Trachtenberg 2008;

Gravier, Klasing, and Moncel 2008; Suomela 2007) were it was shown that the MICS problem cannot be approximated to a better factor than $O(\log n)$, where n denotes the number of vertices in the graph.

2.1.1 Identifying Codes

Throughout the subsequent chapters in the thesis, the Identifying Code problem has been defined, for both undirected and directed graphs, as follows -

Definition 2.1.1. A vertex set V' of an undirected graph $G = (V, E)$ is defined as the Identifying Code Set (ICS) for the vertex set V, if for all $v \in V$, $N^+[v] \cap V'$ is unique where, $N^+[v] = v \cup N(v)$ and $N(v)$ represents the set of nodes adjacent to v in $G = (V, E)$. The *Minimum Identifying Code Set* (MICS) problem is to find the Identifying Code Set of *smallest cardinality.*

The vertices of the set V' may be viewed as *alphabets* of the code, and the *string* made up by the concatenation of the alphabets of $N^+[v] \cap V'$, may be viewed as the unique "code" (or signature) for the node v. This can be better understood with the help of the following example. Consider the graph illustrated in Figure 3. In this graph $V' = \{v_1, v_2, v_3, v_4\}$ is an ICS, as it can be seen from Table 2 that $N^+[v] \cap V'$ is *unique* for all $v \in V$. From the table, it can be seen that the code for node v_1 is v_1, the code for v_5 is v_1, v_2, the code for v_{10} is v_3, v_4, etc. In other words, if node v_5 were to engage in anomalous behavior, then sensors placed at v_1, v_2 will "sense" this anomaly and uniquely identify node v_5.

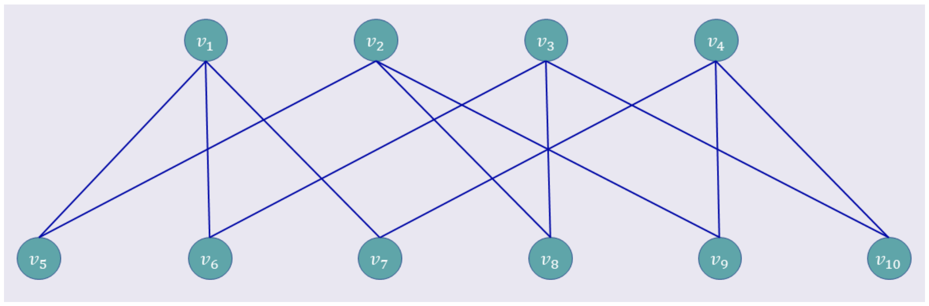


Figure 3. An example undirected graph

$N^+[v_1] \cap V' = \{v_1\}$	$N^+[v_2] \cap V' = \{v_2\}$
$N^+[v_3] \cap V' = \{v_3\}$	$N^+[v_4] \cap V' = \{v_4\}$
$N^+[v_5] \cap V' = \{v_1, v_2\}$	$N^+[v_6] \cap V' = \{v_1, v_3\}$
$N^+[v_7] \cap V' = \{v_1, v_4\}$	$N^+[v_8] \cap V' = \{v_2, v_3\}$
$N^+[v_9] \cap V' = \{v_2, v_4\}$	$N^+[v_{10}] \cap V' = \{v_3, v_4\}$

Table 2. Nodes with unique signatures

Definition 2.1.2. Two nodes $u, v \in V$ are said to be "twins" if $N^+[u] = N^+[v]$ in an undirected graph.

Observation: Identifying Code Set (ICS) of an undirected graph $G = (V, E)$ does not exist, if for any two nodes $u, v \in V$, $N^+[u] = N^+[v]$. This phenomenon is popularly known as "twin" vertices, in standard graph theoretical literature (Charon et al. 2007). In other words, *the necessary and sufficient condition* for an undirected network to have an Identifying Code is that the network be "twin-free".

Definition 2.1.3. A vertex set V' of a directed graph $G = (V, E)$ is defined as the Identifying Code Set (ICS) for the vertex set V, if for all $v \in V$, $N^{out}[v] \cap V'$ is unique where, $N^{out}[v] = v \cup N^{out}(v)$ and $N^{out}(v)$ represents the set of out-neighbors of v in $G = (V, E)$. As before, the *Minimum Identifying Code Set* (MICS) problem is to find the Identifying Code Set of *smallest cardinality.*

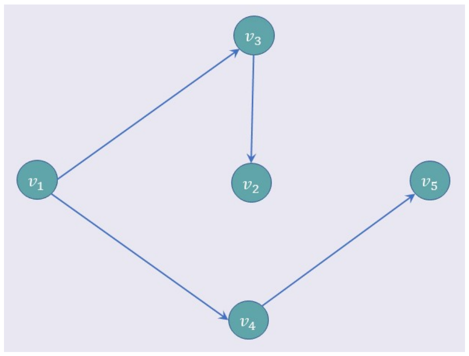


Figure 4. An example directed graph

$N^{out}[v_1] \cap V' = \{v_3, v_4\}$	$N^{out}[v_2] \cap V' = \{v_2\}$
$N^{out}[v_3] \cap V' = \{v_2, v_3\}$	$N^{out}[v_4] \cap V' = \{v_4, v_5\}$
$N^{out}[v_5] \cap V' = \{v_5\}$	

Table 3. Nodes with unique signatures

Definition 2.1.4. Two nodes $u, v \in V$ are said to be "twins" if $N^{out}[v] = N^{out}[u]$ in a directed graph.

Observation: Similar to the case of undirected graphs, Identifying Code Set (ICS) of a directed graph $G = (V, E)$ does not exist, if any two nodes $u, v \in V$ are "twins".

Consider the graph illustrated in Figure 4. In this graph $V' = \{v_2, v_3, v_4, v_5\}$ is an ICS, as it can be seen from Table 3 that $N^{out}[v] \cap V'$ is *unique* for all $v \in V$.

2.1.2 Discriminating Codes

The Discriminating Code problem was first studied in (Charbit et al. 2006; Charbit et al. 2008; Charon et al. 2008) and can be defined as follows -

Definition 2.1.5. Given a bipartite graph $G = (V_1 \cup V_2, E)$, the subset $V_2' \subseteq V_2$, is defined as a Discriminating Code Set (DCS) for the vertex set V_1, if $\forall v \in V_1, N(v) \cap V_2'$

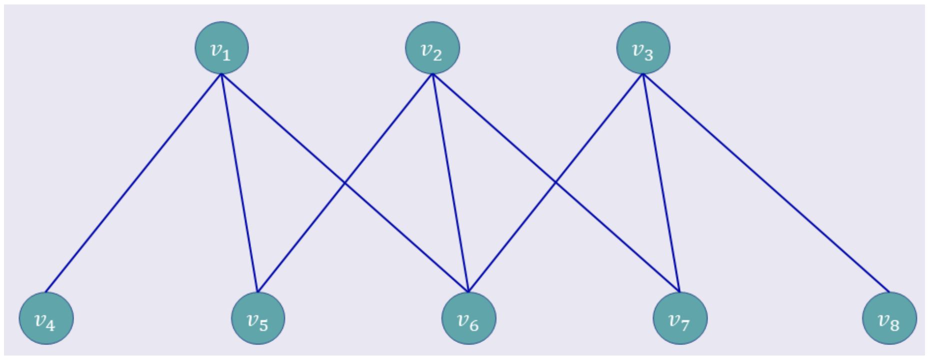


Figure 5. An example bipartite graph

$N(v_1) \cap V_2' = \{v_5\}$
$N(v_2) \cap V_2' = \{v_5, v_7\}$
$N(v_3) \cap V_2' = \{v_7\}$

Table 4. Nodes with unique signatures

is unique where, $N(v) \subseteq V_2$ represents the set of nodes adjacent to $v \in V_1$. The Minimum Discriminating Code Set (MDCS) problem is to find the DCS of smallest cardinality.

Thus, the Discriminating Code problem is a restricted version of the Identifying Code problem, *where the graph is bipartite*. Simply stated, the monitoring problem in such a scenario, is to obtain unique signatures for nodes in V_1 using a subset of the nodes in V_2 as potential locations for sensor placement. We illustrate this with the help of the an example, shown in Figure 5. Here, $V_1 = \{v_1, v_2, v_3\}$, $V_2 = \{v_4, v_5, v_6, v_7, v_8\}$ and the DCS $V_2' = \{v_5, v_7\}$, as $\forall v \in V_1, N(v) \cap V_2'$ is unique, as shown in Table 4.

Observation: Discriminating Code Set (DCS) of a bipartite graph $G = (V_1 \cup V_2, E)$ does not exist if any two nodes $u, v \in V_1$ are "twins". As mentioned previously, the necessary and sufficient condition for a bipartite network to have a Discriminating Code is that the network be "twin-free".

2.1.3 Graph Coloring with Seepage

The MICS and MDCS computation problem can be viewed as a variation of the standard Graph Coloring problem. We will refer to this version as the *Graph Coloring with Seepage (GCS)* problem. In the standard graph coloring problem, when a color is *assigned* (or injected) to a node, only that node is colored. The goal of the standard graph coloring problem to use as few distinct colors as possible such that (i) every node receives a color, and (ii) no two adjacent nodes of the graph have the same color. In the GCS problem, when a color is assigned (or injected) to a node, not only that node receives the color, the color also *seeps* into all the adjoining nodes. As a node v_i may be adjacent to two other nodes v_j and v_k in the graph, if the color red is injected to v_j, not only will v_j become red, but also v_i will become red as it is adjacent to v_j. Now if the color blue is injected to v_k, not only will v_k become blue, but also, the color blue will seep in to v_i as it is adjacent v_k. Since v_i was already colored red (due to seepage from v_j), after color seepage (blue) from v_k, its color will be a *combination of red and blue, i.e., purple.* At this point all three nodes v_i, v_j and v_k have a color and all of them have distinct colors (purple, red and blue respectively). The goal of the GCS problem is to inject colors to as few nodes as possible, such that *(i) every node receives a color, and (ii) no two nodes of the graph have the same color.*

Suppose that the node set V' is an ICS of of a graph $G = (V, E)$ and $|V'| = p$. If p distinct colors are injected to the nodes of V' (one distinct color to one node of V'), then as by the definition of ICS for all $v \in V$, if $N^+[v] \cap V'$ is unique, all nodes of $G = (V, E)$ will be colored and no two nodes will have the same color. Accordingly, computation of the MICS problem is equivalent to computation of the GCS problem.

Similar arguments also hold for computation of the MDCS problem. Assume that the node set V_2' is a DCS of of a graph $G = (V_1 \cup V_2, E)$ and $|V_2'| = p$. In this case, if p distinct colors are injected to the nodes of V_2' (one distinct color to one node of V_2'), then as by the definition of DCS for all $v \in V_1$ if $N(v) \cap V_2'$ is unique, all nodes $v \in V_1$ will be colored and no two nodes will have the same color.

To compute the MICS using GCS, we must first understand the subtle difference between the flow of "active" signals (or information) and color seepage. The direction of color seepage is *opposite* to the direction of the flow of "active" signals (or information). For instance, consider the undirected graph in Figure 3. Following the GCS methodology, colors are injected in nodes v_1, v_2, v_3, v_4 (deploying law enforcement agents to monitor these individuals). The injection of colors in these four nodes ensures that all the ten nodes in the graph receive a *unique* color via seepage, as is observed in Table 2. For example, colors injected at nodes v_1, v_2 seep into v_5, color injected at v_3 seeps into v_6, v_8, v_{10}, and so on, resulting in unique colors for all the nodes in the graph. Now, if v_5 becomes "active", then the signal of such activity would flow to *all* the neighbors of v_5. In other words, the neighbors of v_5 would become aware of v_5's activity (which includes v_1, v_2), and v_5 would be uniquely identified, when the signal reaches nodes v_1 and v_2.

In the case of a directed graph, the direction of the edge signifies the direction of "active" signal flow, and as before, the direction of color seepage is *opposite* to that of signal flow. For instance, consider the directed graph in Figure 4. The MICS is v_2, v_3, v_4, v_5. The activity signal of v_1 would flow from v_1 to v_3 and v_4 (the out-neighbors of v_1). Colors injected at nodes v_3 and v_4 would seep into node v_1, and thus nodes v_3 and v_4 would enable the unique identification of v_1, from which the signal originated. The unique identification of all the other nodes in the graph can be seen in the Table 3.

2.1.4 Transformation to Minimum Hitting Set Problem

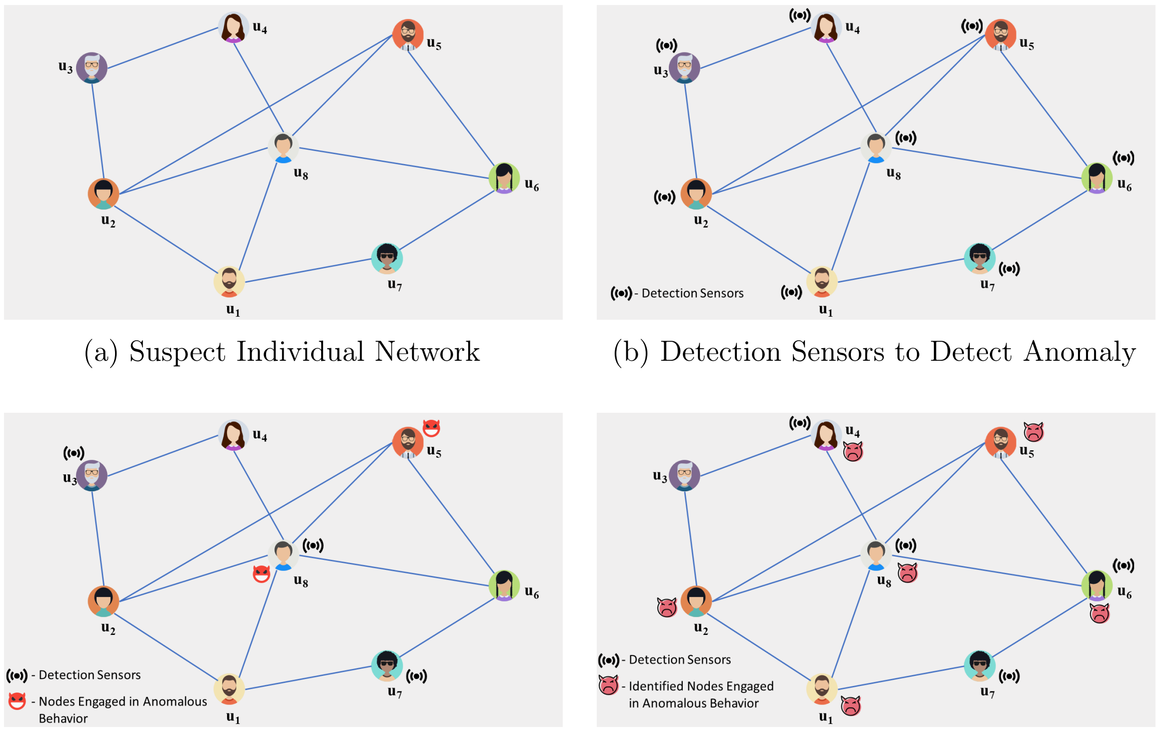
Apart from the GCS viewpoint discussed previously, we can also view the MICS problem as a Minimum Hitting Set (MHS) problem. This transformation allows us to utilize the well known greedy algorithm of the MHS problem in order to provide the approximation bound for the MICS problem. Interestingly, previous research efforts which determined the approximation bounds for the MICS problem explored convoluted routes, such as determining entropy, disjoint sets, etc. (Xiao, Hadjicostis, and Thulasiraman 2006; Gravier, Klasing, and Moncel 2008). The greedy MHS approximation algorithm, on the other hand, is easier to understand and implement. It may be noted that the greedy heuristic for the HS problem provides a $O(log\ m)$ factor performance bound, where m is the number of elements in the collection set (Vazirani 2013). In the following, we define the minimum HS problem -

Definition 2.1.6. Given a universal set $\mathcal{U} = \{u_1, ..., u_n\}$, and a collection set $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_m\}$, where $\mathcal{S}_i \in \mathcal{U}$, find the smallest subset $U' \subseteq \mathcal{U}$, which hits every set $\mathcal{S}_i \in \mathcal{S}$.

Definition 2.1.7. Closed Neighborhood of $v_i = CN(v_i) = N^+(v_i)$, where $N^+(v_i) = N(v_i) \cup \{v_i\}$, where $N(v_i)$ denotes the neighborhood of the node v_i.

Definition 2.1.8. Distinguishing Set for v_i and $v_j = DS(v_i, v_j) = CN(v_i) \bigoplus CN(v_j)$. $\bigoplus$ denotes the symmetric difference operation between the closed neighborhood sets $CN(v_i)$ and $CNv(j)$. In other words, picking at least one element from the set $DS(v_i, v_j)$ will distinguish between nodes v_i and v_j.

Definition 2.1.9. Universal Set $\mathcal{U} = \{v_1, ..., v_n\}$, where each element v_i is a node in the complex network graph and Collection Set $\mathcal{S} = \cup_{i=1}^{n}[CN(v_i)\ \cup_{j=1}^{n} \{DS(v_i, v_j)\}]$.


u3
u4
u5
u8
u2
u6
u1
u7
(•) - Detection Sensors
- Nodes Engaged in Anomalous Behavior
- Identified Nodes Engaged in Anomalous Behavior

(a) Suspect Individual Network

(b) Detection Sensors to Detect Anomaly

(c) Set Cover Based Covering Fails to Distinguish Between Users u_5 and u_8

(d) Identifying Code Based Covering Distinguishes Each Node in The Graph

Figure 6. Identifying Code Overcoming The Limitations of Set Cover

Our objective is to select the minimum number of elements from $\mathcal{U}$, such that all the elements in $\mathcal{S}$ are hit. Hitting all the elements in $\mathcal{S}$ ensures that, (i) all $CN(v_i)$ sets are hit, which in turn ensures that all nodes in the graph are monitored (a detection sensor has been placed in the closed neighborhood of v_i), and (ii) all $DS(v_i, v_j)$ sets are hit, which in turn ensures that all the nodes in the graph are *uniquely monitored*. Thus the computation of this variant of the minimum HS problem is equivalent to solving the MICS problem.

2.2 Why Identifying Codes?

The research objective of this thesis is to determine a subset of the nodes of the graph, on which detection sensors can be placed, using which we can uniquely monitor all the nodes in the graph for anomalous behaviors. Essentially, our problem is a coverage problem and in order to *uniquely cover* all the nodes in the graph, we need to optimize the locations (or nodes) where we can deploy or place the detection sensors, i.e., in other words, sensor placement optimization. Numerous studies on sensor placement optimization problem follow the Set Cover (SC) formulation to find a solution (Wang 2011; Cardei and Wu 2006; Kleinberg and Tardos 2006). The SC approach in this domain can be modeled as follows

Definition 2.2.1. The universal set $\mathcal{U} = \{u_1, u_2, ..., u_n\}$, in which each u_i corresponds to a node in the graph. The collection set $\mathcal{S} = \{\mathcal{U}'_1, \mathcal{U}'_2, ..., \mathcal{U}'_n\}$ contains n subsets of the universal set where each subset $\mathcal{U}_i$ corresponds to the closed neighborhood of the node i (closed neighborhood includes the node i in its neighborhood set). The objective here is to select a subset $\mathcal{S}' \subseteq \mathcal{S}$ of minimum cardinality, which covers all the elements in the universal set.

Consider the following example, where law enforcement agents are tracking a given network of suspect individuals. A human-human interaction network, with 8 individuals is illustrated in Figure 6a. Edges signify friendship. The objective here is to monitor *all* the 8 individuals, and identify those engaging in suspected terror related activities. One trivial approach to realize this goal is to monitor the behavior of all the individuals, as shown in Figure 6b. As mentioned previously, for an individual u_i in the network, if u_i were to become active in terror related activities, its friends/associates would have some idea about it. Generally speaking, if a node in the network were to behave anomalously, then it's neighborhood would be "aware" of this anomaly. Moreover, only *one* individual can become active in anomalous activities at a given time step.

In this example, it can be verified that if the law enforcement agency monitoring or tracking this network, deploys detection sensors (law enforcement agents) on individuals u_3, u_7, u_8 (an optimal set cover for this graph), all the individuals can be monitored for anomalous behavior by least one sensor (agent), as is illustrated in Table 5. In Table 6, we present the sensors that are actually monitoring the individuals $u_1 - u_8$, an inverse of Table 5. The drawback of the SC approach for the optimal sensor placement problem is that, it may fail to *uniquely identify* an individual, who initiated the anomalous behavior. From Figure 6c it is evident that if individual u_5 were to become active in anomalous behavior, then the sensor placed on u_8 would be activated as, users u_5 and u_8 are friends with each other. Additionally, it is trivial to note that if u_8 behaves anomalously, then the sensor placed at u_8 will also get triggered. Note that, the sensor gets triggered if, (i) u_5 engages in anomalous behavior, (ii) u_8 engages in anomalous behavior, and (iii) both u_5 and u_8 engage in anomalous behavior. In other words, following the SC approach and placing detection sensors on u_3, u_7, u_8, the law enforcement agency *will not* be able to *uniquely distinguish* between u_5 and u_8. More such violations (denoted by *) are presented in Table 6. It can be seen that u_1 and u_6 are sensed by sensors placed at u_7 and u_8, and u_2 and u_4 are sensed by sensors placed at u_3 and u_8. The implication of this is that if sensors u_7 and u_8 get triggered, then u_1 and u_6 cannot be distinguished.

Sensor Location	Individuals Sensed
u_3	u_2, u_3, u_4
u_7	u_1, u_6, u_7
u_8	$u_1, u_2, u_4, u_5, u_6, u_8$

Table 5. Sensors Covering Individuals

Individuals Sensed	Sensor Location	Individuals Sensed	Sensor Location
u_1	$u_7, u_8{}^*$	u_5	$u_8{}^{***}$
u_2	$u_3, u_8{}^{**}$	u_6	$u_7, u_8{}^*$
u_3	u_3	u_7	u_7
u_4	$u_3, u_8{}^{**}$	u_8	$u_8{}^{***}$

Table 6. Individual Monitoring by Set Cover Approach

Sensor Location	Individuals Sensed
u_4	u_3, u_4, u_8
u_6	u_5, u_6, u_7, u_8
u_7	u_1, u_6, u_7
u_8	$u_1, u_2, u_4, u_5, u_6, u_8$

Table 7. Sensors Covering Individuals

This failure to uniquely identify an individual, can be overcome by deployment of additional detection sensors. Suppose that detection sensors are now deployed to u_4, u_6, u_7, u_8 instead of u_3, u_7, u_8. Again, this placement covers all of the nodes in the graph, as is illustrated in Table 7. Now, if u_5 or u_8 engage in anomalous behavior, then their behavior would disseminate to sensors placed at u_8 and u_4, u_6, u_8 respectively. Thus, the law enforcement agency can now *uniquely distinguish* between the two individuals. This new sensor deployment strategy ensures that the violations listed in Table 6 can be overcome, and can be verified with the help of Table 8. Observe that the deployment of four sensors, instead of three, alleviates the problem encountered

Individuals Sensed	Sensor Location	Individuals Sensed	Sensor Location
u_1	u_7, u_8	u_5	u_6, u_8
u_2	u_8	u_6	u_6, u_7, u_8
u_3	u_4	u_7	u_6, u_7
u_4	u_4, u_8	u_8	u_4, u_6, u_8

Table 8. Individual Monitoring by Identifying Codes

in the previous deployment, by ensuring unique identification of individuals behaving anomalously, as illustrated in Figure 6d.

To conclude, Identifying Codes based monitoring strategies deploy more sensors than its Set Cover counterpart. This not only ensures monitoring coverage of the entire network, but also *unique* coverage, which allows complex network operators to uniquely identify nodes in the network, who initiate anomalous behavior.

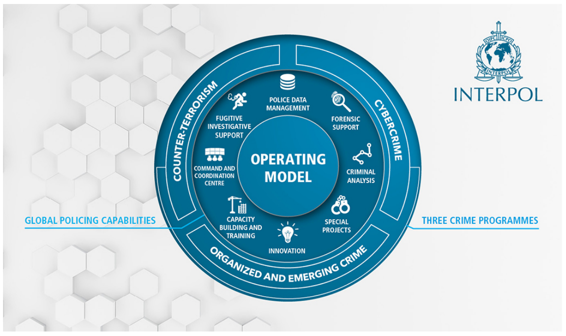
Chapter 3

HUMAN-HUMAN INTERACTION NETWORKS

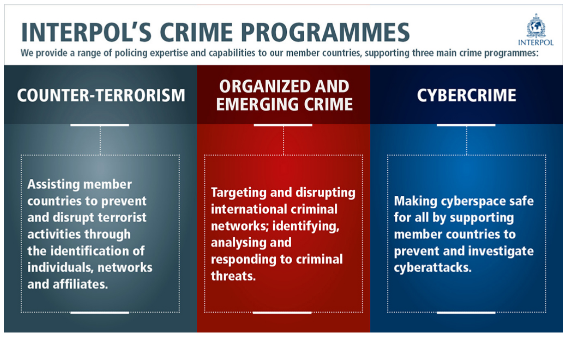
The past couple of decades has witnessed an unprecedented rise in organized crime. This rise, coupled with increasing intricacies of organized crime, poses significant and evolving challenges for international law enforcement authorities. With the passage of time, authorities such as Interpol, have discovered that modern criminal organizations have adopted a networked structure, a shift away from the traditional hierarchical structure. Fluid networked structures make it difficult for the authorities to apprehend individuals associated with each network, and consequently, to disrupt the operations of the network. Various research groups have analyzed prison/courtroom transcripts, to create an organizational structure of known individuals, or a social network of individuals, suspected to be a part of a major drug/terrorist organization. These social networks have been studied fairly extensively from network centrality perspectives, to understand the role of suspect individuals in the network. With drug and terror offenses increasing globally, the list of suspect individuals has also been growing over the past decade. As it takes significant amount of technical and human resources to monitor a suspect, an increasing list entails higher resource requirements on the part of the authorities, and monitoring all the suspects soon becomes an impossible task. In this chapter, the primary focus is on two types of networks - (i) Drug Trafficking Organizations (DTOs), and (ii) Terrorist Organizations (TOs).

In this chapter, we elaborate on five novel sub-problems pertaining to the problem of uniquely monitoring such networks. We first show how Identifying Codes can be utilized for monitoring such networks as well as achieve significant reduction in resources (for e.g., the number of agents deployed to monitor the network). Next, we address the problem of sensor failure and present results for k-Fault Tolerant Identifying Codes. Recall that, one assumption made thus far, was that we only considered single node activations at a given instance of time. In other words, the scenario of simultaneous multiple nodes being active in anomalous behavior was not considered. Here, we present approaches for the case where *two nodes* can simultaneously become active in anomalous behavior. Prior works focused on using graph centrality metrics as a means to monitor the "important" individuals in the network. In this chapter, we show how Identifying Codes based monitoring strategies are superior to those based on the standard graph centrality metrics. Finally, we present the Augmented Identifying Code problem, which follows from the standard Identifying Code problem, and which is more realistic in nature.

The International Criminal Police Organization, or more popularly known as Interpol, is an inter-governmental organization established to counter three broad types of transnational organized crime, as illustrated in Figure 7a and Figure 7b. Organized criminal networks are usually involved in multiple types of criminal activities, spanning several countries, such as drug trafficking, terrorism, human trafficking, counterfeiting, etc. It is widely known that organized criminal organizations flourish due to their significantly high revenues since their criminal businesses resemble those of legitimate international businesses (Interpol, n.d.). The ultimate goal of such organizations is to generate the maximum profits with minimum risks. On the other hand, the objective of Interpol is to disrupt the operations of such organizations. This task is far from trivial, as Interpol agencies conduct significant amount of criminal analysis based on numerous sources of intelligence, in order to link different organizational members - financiers, recruiters, distributors, etc. and create an organizational network. Furthermore,



(a) Interpol Model



INTERPOL'S CRIME PROGRAMMES

We provide a range of policing expertise and capabilities to our member countries, supporting three main crime programmes:

COUNTER-TERRORISM	ORGANIZED AND EMERGING CRIME	CYBERCRIME
Assisting member countries to prevent and disrupt terrorist activities through the identification of individuals, networks and affiliates.	Targeting and disrupting international criminal networks; identifying, analysing and responding to criminal threats.	Making cyberspace safe for all by supporting member countries to prevent and investigate cyberattacks.

(b) Interpol's Broad Criminal Divisions

Figure 7. International Criminal Police Organization

additional human resources (field agents) are required by the agencies, to monitor these organizational members (Fooner 1985).

The United Nations Office on Drugs and Crime (UNODC), a partner of Interpol, is a global authority in the fight against illicit drugs and international crime. The UNODC defines drug trafficking as *"a global illicit trade involving the cultivation, manufacture, distribution and sale of substances, which are subject to drug prohibition laws"* (UNODC, n.d.). About 450,000 people died as a consequence of drug use in 2015, according to the World Health Organization. About 168,000 of these deaths were directly associated with drug overdoses. The 2018 World Drug Report released by the UNODC (UNODC 2018) continued to highlight several grave statistics pertaining to global drug use. 5.6% of the global population belonging to the 15-64 age group (about 275 million people worldwide), consumed drugs *at least once* in 2016. The 15-16 age group (almost 14 million individuals) was responsible for the highest consumption of Cannabis globally.

These *recorded* numbers, in a way, are indicative of the ever growing demand for illicit drug consumption, and various Drug Trafficking Organizations (DTOs) are responding by increasing the production rates of their respective drugs. To counter the activities of DTOs, law enforcement agencies are spending significant amounts of money to just monitor individuals associated with DTOs. The US alone spends about $40 billion a year investigating drug offenses (Law 2013). The investigation of drug offenses usually generates a list of suspects. With drug offenses increasing by the passing year, the suspect database also continues to grow. As a result, the amount of technical and human resources required to monitor a suspect in the database also grows. After a certain point of time, monitoring all the suspects in the database may become an impossible task.

On the other hand, global terrorism is a major source of concern for not only Interpol, but also national level and state level governments as well. The Interpol Terror Watch List is one such database keeping track of terrorist organizations and individuals involved with these organizations. Similar to monitoring drug networks, actively monitoring such organizational networks is a considerable challenge on the part of law enforcement agencies in the sense that, significant amount of resources must be utilized to monitor these networks. Oftentimes, the law enforcement agencies do not have sufficient resources to monitor these organizations and their members effectively. It has been reported that the French Centre for the Analysis of Terrorism has determined that it takes as many as 20 agents per suspect to conduct 24-hour surveillance. On multiple incidences of terrorist attacks in recent times across Europe, it has been observed that the perpetrators of the attack were in the suspect databases of the law enforcement authorities, but weren't under active surveillance at the time of the attack due to resource limitations on the part of the authorities (Bernstein 2017). The news organization Politico (Cooper 2016), reported in October 2016 that the French authorities were monitoring around 15,000 individuals who were suspected of being radical Islamists. The Politico report was based on an earlier publication in the French journal, La Journal du Dimanche. The ABC news affiliated TV station WJLA in Washington D.C., reported in 2017 that, the list has tripled over the last two years (Bernstein 2017). The database is managed by France's Counter-Terrorism Coordination Unit. Obviously, the resources and manpower needed to keep all the terror suspects under surveillance is enormous and often are way beyond the available resources of any local law enforcement authority.

According to reports, the two suspects in the attack against French police on April 20, 2017, on the Champs-Elysees, were known to French anti-terrorism authorities. The November 2015 attacks in Paris that claimed a total of 130 lives, involved a small network of ISIS-linked terrorists in France and Belgium. Of the 10 individuals involved, several were known to authorities. When 12 people were killed at the Paris headquarters of Charlie Hebdo, a satirical magazine, all three of the terrorists had been under close watch. Cherif Kouachi, Said Kouachi and Amedy Coulibaly were under police surveillance for three years, but eventually dropped in the summer of 2014, only months before the deadly January 2015 attack (Wikipedia 2022).

That being said however, over the past couple of decades, law enforcement agencies have managed to disrupt the organization of numerous DTOs/TOs, by deploying various surveillance strategies such as wiretaps, cameras, GPS trackers, etc. (Law 2013). Authorities analyzed transcripts obtained from the interrogation of suspect individuals, to identify key actors and their relationships, to create a social network (Law 2013; Natarajan 2000, 2006). These constructed social networks have been extensively studied by researchers, utilizing standard centrality metrics, to realize particular objectives such as, to determine the importance of network members, presence of sub groups, number and nature of key facilitators, etc. In this chapter, we propose novel Drug Network Monitoring (DNM) and Terror Network Monitoring (TNM) approaches, based on the mathematical concept of Identifying Codes and its extensions, that not only reduces resource requirements on the part of law enforcement agencies (reduction in the number of agents required to monitor these organizations), but also provides the capability of uniquely identifying a suspect, when the suspect becomes "active" in drug trafficking related activities. Our approach has the following assumption: when an individual in a DTO/TO becomes "active" in drug trafficking related activities, his/her friends/associates will have some knowledge of the individual's plan. Accordingly, even if the individual is not under direct surveillance by the law enforcement authorities (recording phone calls, movement, social interactions with other individuals), but the individual's friends/associates are, then *the individual involved in drug related activities can be uniquely identified.* Most importantly, we compare our approach with other approaches utilizing standard centrality metrics, and show that standard centrality based approaches- *(i) do not necessarily guarantee unique identification of every individual, and (ii) lead to a wastage of resources on the part of the authorities.*

3.1 Related Work

In the past few years, significant research on DTO and TO networks have been conducted utilizing Social Network Analysis (SNA). Fooner in (Fooner 1985), was one of the first to illustrate the networked structure of organized crime and highlighted the changes in the behaviour of the network, with the advancement of technology. Natarajan in (Natarajan 2000), analyzed wiretap data to create an organizational structure and studied the roles of particular individuals. She analyzed over 2000 wiretap conversations and performed SNA of phone contacts, to reveal a large and loosely structured group of 294 individuals in (Natarajan 2006). Bright and Delaney in (Bright, Hughes, and Chalmers 2012), utilized SNA to study the evolution of a drug trafficking network, based in Australia. They observed changes in centrality scores and the roles performed by particular individuals. (Bright and Delaney 2013) utilized individual attributes coupled with centrality measures to identify key actors in a drug trafficking network. (Bright et al. 2015) analyzed judges' sentencing comments to create a network of individuals involved in the distribution of methamphetamine in Australia during the 1990s. (Heber 2009) studied the network of drug offenders in Sweden to analyze the types of criminal activity they were involved. (Hughes, Bright, and Chalmers 2017) analyzed court transcripts and identified key actors to create a social network. They analyzed this social network to explore product diversification in three drug syndicates in Australia. (Ressler 2006) investigated the suitability of social network analysis for studying terrorist networks. (Krebs 2002) mapped the 9/11 terror network from articles in leading newspapers. (Carley, Lee, and Krackhardt 2002) and (Carley et al. 2003) explored the potential of using social network analysis and multi-agent modeling for the purpose of destabilizing terrorist networks. (Fu et al. 2014) studied the bipartite networks of terrorist organizations in East Turkestan. (Borgatti 2006) showed the ineffectiveness of the standard centrality metrics in identifying "key actors" in terror networks and introduced two new metrics Key Player Problem Negative (KPP-Neg) and Key Player Problem Positive (KPP-Pos) and showed that these two were superior to the standard centrality measures.

In addition to research on DTOs and TOs through SNA, the last few years have seen a significant amount of research on Identifying Codes and its applications to networks. Karpovsky *et. al.* introduced the concept of Identifying Codes in (Karpovsky, Chakrabarty, and Levitin 1998) and provided results for Identifying Codes for graphs with specific topologies, such as binary cubes and trees. Using

Identifying Codes, Laifenfeld *et. al.* studied joint monitoring and routing in wireless sensor networks in (Laifenfeld and Trachtenberg 2008). (Irene Charon, Hudry, and Lobstein 2002; Irène Charon, Hudry, and Lobstein 2003) studied complexity issues related to computation of minimum Identifying Codes for graphs and showed that in several types of graphs, the problem is NP-hard. (Ray et al. 2004) generalized the concept of Identifying Codes, to incorporate robustness properties to deal with faults in sensor networks. Suomela and Gravier, in (Suomela 2007; Gravier, Klasing, and Moncel 2008) determined that the Identifying Code problem had a $O(\log n)$ - approximation factor.

A special case, where only a subset of nodes needs a unique code, can be modeled with a bipartite graph, and this version of Identifying Codes is called “Discriminating Codes” and was studied in (Charbit et al. 2006; Charbit et al. 2008). This special case is relevant for our study as, our problem formulation of bipartite networks requires us to find the unique signatures of all the nodes in one side of the bi-partition of a bipartite graph, by selecting only a subset of the nodes in the other side of bi-partition. This formulation corresponds directly to the “Discriminating Codes” problem.

It may be noted that Drug and Terror related SNA studies have usually utilized network centrality measures to *identify key actors in the network* (Bright and Delaney 2013; Gialampoukidis et al. 2016; Berzinji, Kaati, and Rezine 2012). However, in this chapter, our objective is different in the sense that, we aim to *uniquely identify every actor in the network, should they become active in drug/terror related activities*, and not just the “key actors”. Additionally, we show that if popular centrality metrics were utilized for realizing our objective, then the law enforcement agencies would end up wasting their resources. In other words, simply monitoring “important” individuals would lead to a wastage in resources.

3.2 Utilizing Identifying Codes for Resource Reduction

We formalize both the Drug Network Monitoring (DNM) and Terrorist Network Monitoring (TNM) problems, utilizing three types of networks - undirected, directed and bipartite. It may be noted that, for the three types of networks, the nodes are either individuals or organizations (based on the dataset), and the edges represent the relationship between the individuals or organizations.

3.2.1 Problem Formulation

If the network can be represented as an undirected graph $G = (V, E)$, the objective is to monitor (surveillance of the suspected individual) a subset of the nodes $V' \subseteq V$, so that each and every node $v \in V$, can be *uniquely* identified, in case the individual represented by v becomes active in drug/terror related activities. As mentioned previously, we assume that only *one* node v becomes active at a time and each and every node $v \in V$ can be monitored.

Formally stated, the objective of undirected networks version of DNM and TNM is to monitor the least number of suspect individuals, in order to *uniquely* identify any one of the suspect individuals, should they become active in drug trafficking or terror related activities. Recalling the objective of the GCS problem described in chapter 2, a natural connection between the GCS problem and the DNM and TNM problems can be made. The suspect individuals who have to be monitored, can be thought of as nodes in the network, where colors have to be injected in the GCS problem. In the GCS problem, injection of colors in a subset of nodes in the network (or graph), ensures that each node in the network receives a unique color. Thus, monitoring (assigning law enforcement authorities to) this subset of suspect individuals will result in the unique identification (or signature) for all of the suspect individuals (or all the nodes) in the network. In other words, the unique identification of a suspect individual (or node) corresponds to the unique color associated with that individual (or node). Note that, in chapter 2, the GCS problem was shown to be equivalent to the computation of the MICS problem. By transitivity, it is evident that the undirected DNM and TNM problems are equivalent to solving the MICS of the corresponding undirected DTO or TO network.

As is the case for undirected graphs, the objective for directed graphs is to monitor the subset of the nodes $V' \subseteq V$, so that each and every node $v \in V$ can be *uniquely* identified, in case the individual represented v becomes active in drug or terror related activities. Additionally, we assume that only *one* node can become "active" at a time. Since the objective here is similar to that of the undirected networks scenario, similar equivalences can be constructed for the directed networks and it can be easily shown that the directed DNM and TNM problems are equivalent to solving the MICS of the corresponding directed DTO or TO network.

For bipartite drug/terror networks, represented by $G = (V_1 \cup V_2, E)$, V_1 represents the set of individuals who may become "active" in drug or terror related activities, and V_2 represents the set of individuals who can be actually monitored by law enforcement authorities. The objective of this version of the problem is to inject colors in the fewest number of nodes in V_2, such that every node in V_1 receives a unique color. Oftentimes, assigning law enforcement authorities to monitor certain individuals in the drug or terror networks may be difficult (for instance, monitoring individuals belonging to the higher echelons of the organization), and this set of individuals is denoted by the node set V_1. In such scenarios, it may be easier to assign authorities to and monitor the individuals who are in contact with individuals denoted by nodes in V_1. In other words, only nodes in V_1 may become "active" and can be monitored by monitoring nodes in V_2, in contrast to the previous two subsections, where it was assumed that any node $v \in V$, may become active and can be monitored. It is evident that the bipartite DNM and TNM problems are equivalent to solving the MDCS of the corresponding DTO or TO network.

3.2.2 Problem Solution

In this section, we provide solution techniques for the three different types of networks, utilizing Integer Linear Programs (ILPs).

Instance: $G = (V, E)$, an undirected graph.

Problem: Find the smallest subset $V' \subseteq V$, such that injection of colors at these nodes, ensures that each node $v \in V$, receives a unique color (atomic/composite) through seepage.

We use the notation $N^+[v_i]$ to denote the closed neighborhood of v_i, $\forall v_i \in V$. Corresponding to each $v_i \in V$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{v_i \in V} x_i$

Coloring Constraint: $\sum_{v_i \in N^+[v_j]} x_i \geq 1$, $\forall v_j \in V$

Unique Coloring Constraint:

$\sum_{v_i \in \{N^+[v_j] \bigoplus N^+[v_k]\}} x_i \geq 1$, $\forall v_j \neq v_k, \in V$

$N^+[v_j] \bigoplus N^+[v_k]$ denotes the Exclusive-OR of the node sets $N^+[v_j]$ and $N^+[v_k]$. It may be noted that the objective function ensures that the fewest number of nodes in V are assigned a color. The Coloring Constraint ensures that every node in V receives at least one color through seepage from the colors injected at nodes in its closed neighborhood. A consequence of the Coloring Constraint is that, a node in V may receive more than one color through seepage from the colors injected at its neighborhood. The Unique Coloring Constraint ensures that, for every pair of nodes (v_j, v_k) in V, at least one node in the node set $N^+[v_j] \bigoplus N^+[v_k] \subseteq V$ is injected with a color. This guarantees that v_j and v_k will not receive identical colors.

Instance: $G = (V, E)$, a directed graph.

Problem: Find the smallest subset $V' \subseteq V$, such that injection of colors at these nodes, ensures that each node $v \in V$, receives a unique color (atomic/composite) through seepage.

We use the notation $N^{out}[v_i]$ to denote the closed out-neighborhood of v_i, for any $v_i \in V$. Corresponding to each $v_i \in V$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{v_i \in V} x_i$

Coloring Constraint: $\sum_{v_i \in N^{out}[v_j]} x_i \geq 1$, $\forall v_j \in V$

Unique Coloring Constraint:

$\sum_{v_i \in \{N^{out}[v_j] \bigoplus N^{out}[v_k]\}} x_i \geq 1, \forall v_j \neq v_k, \in V$

The objective function and constraints for this ILP are almost identical to the one presented in the previous subsection.

Instance: $G = (V_1 \cup V_2, E)$, an undirected bipartite graph.

Problem: Find the smallest subset $V_2' \subseteq V_2$, such that injection of colors at these nodes, ensures that $\forall v_i \in V_1$, receives a unique color (atomic/composite) through seepage.

We use the notation $N(v_i)$ to denote the neighborhood of v_i, for all $v_i \in V_1 \cup V_2$. Corresponding to each $v_i \in V_2$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{v_i \in V_2} x_i$

Coloring Constraint: $\sum_{v_i \in N(v_j)} x_i \geq 1, \forall v_j \in V_1$

Unique Coloring Constraint:

$\sum_{v_i \in \{N(v_j) \bigoplus N(v_k)\}} x_i \geq 1, \forall v_j \neq v_k, \in V_1$

The notations and their explanations in this ILP is identical to the earlier two ILPs. It can be easily verified that the solution to the ILP finds the smallest subset $V_2' \subseteq V_2$, such that injection of colors at these nodes, ensures that each node $v_i \in V_1$, receives a unique color.

3.2.3 Experimental Results and Discussions

To highlight the effectiveness of our algorithms in the reduction of resources without compromising the ability of unique identification of a suspect, we executed the ILPs on various real world drug and terror network datasets (UCINET 2022). It may be recalled that for a network to have an Identifying Code, it must be “twin-free”. In other words, the necessary and sufficient condition for a network to have an ICS is that the network be “twin-free”. For “twin-free” networks $G = (V, E)$, one trivial Identifying Code set solution is the node set V, although, V may not be the Identifying Code set of *minimum cardinality.* This implies that, if monitors were placed on every node in the network, then all the nodes in the network would receive a unique identification. However, our algorithms show that unique identification for all the nodes in the network can be obtained by deploying agents to monitor a subset $V' \subseteq V$. Since no additional benefits are realized by deploying additional agents, there is absolutely no reason to deploy a larger number of monitors. In Table 9, Table 10 and Table 11, we highlight on the reduction in resource requirements brought about by our methods. As monitoring suspect individuals in drug networks can be a costly affair on the part of law enforcement authorities (Law 2013), a significant reduction in resources will be of great interest to the respective authorities.

3.2.3.1 Datasets

The datasets used for evaluating the efficacy of our approach were obtained from (UCINET 2022). In each dataset, the nodes represent individuals and the edges represent the relationship between these individuals. Natarajan in (Natarajan 2000) analyzed wiretaps and created a social network of Cocaine dealers in New York City. The second dataset, Cocaine Smuggling, contains four social networks unearthed by four investigative operations carried out by law enforcement agencies, involved in smuggling Cocaine from Colombia to Spain. Operation Mambo identified 31 suspect individuals based in Colombia, Operation Juanes identified 51 suspect individuals based in Mexico and Operations Jake and Acero identified 38 and 25 suspect individuals respectively, based in Madrid. The third dataset, Drug Net is a social network of 294 drug users in Hartford. (Natarajan 2006) uncovered a Heroin trafficking organization based in New York City consisting of 38 suspect individuals, which is the fourth dataset. The fifth dataset is the Montreal Street Gang dataset, reconstructed from the drug-distribution operations in Montreal North and contains 35 organizations. Our final dataset is a bipartite network of athletic trainers/health clinics and their associations with MLB athletes, based on the Mitchell Report (Mitchell 2007), which uncovered the illegal uses of steroids in the sport of baseball.

The first terror network on which we applied our technique is the network of the individuals involved in the terror attack in Paris in November 2015. This network consisted of 10 individuals. The Rizal Day bombings of 2000, which occurred in Manila, Philippines, forms our second network. 16 individuals were involved in this attack. The third terror network in our study is the IS-Europe network (or the Zerkani Network), consisting of 39 individuals. The Madrid train bombings network of 2004 is the fourth network in our analysis. The fifth network is the Noordin Mohammed Top network. (Liebig and Rao 2014) identified four individuals, Noordin Top, Azhari Husin, Purnama Putra and Ahmad Ridho, as the key individuals of this network. In our analysis, we view these four individuals as layer 1 individuals, and the remaining as layer 2 individuals. As monitoring the key players (layer 1 individuals) is often significantly more difficult than monitoring layer 2 individuals, we focus on monitoring the layer 1 individuals in an indirect manner, by directly monitoring layer 2 individuals who are interacting with the layer 1 individuals. To capture this mechanism, we construct a bipartite graph $G = (V_1 \cup V_2, E)$, where the nodes in V_1 represent the individuals in layer 1 and the nodes in V_2 represent the individuals in layer 2. An edge $e \in E$ connects a node $u \in V_1$ with a node $v \in V_2$, if u interacts with v. In this network $|V_1| = 4$ and $|V_2| = 15$. We next analyze a network of terrorist organizations and their members, based in East Turkestan. This is a bipartite graph $G = (V_1 \cup V_2, E)$, where V_1 represents the organizations and V_2 represents the individuals. An edge $e \in E$ connects a node $u \in V_1$ with a node $v \in V_2$, if v is a member of u.

3.2.3.2 Results

In this subsection, we present the results of our analyses on undirected and directed drug networks. It may be noted that some networks in our study were not initially "twin-free". Drug networks such as Juanes, Mambo, Heroin and DrugNet, and terror networks such as Philippines, IS-E Zerkani and East Turkestan contained twins. By combining the nodes which form "twins" into super-nodes, one can ensure that the network becomes "twin-free" and subsequently ensure the computation of Identifying Codes. If the nodes $u, v \in V$ form "twins" we can create a super node (u, v) by condensing (combining) nodes u and v. However, as the modified network does not contain either u or v (it has the super node (u, v)), if the individual corresponding to nodes u or v were to become "active", Identifying Code will not be able to distinguish between these two individuals. Hence, further (lower level) analysis will be needed to find out whether node u or v is in the process of being "active". All six networks

Drug Networks	Network Type	Number of Nodes	MICS Cardinality	Reduction in Resources
Operation Juanes	Undirected	50	22	**56%**
Operation Acero	Undirected	25	13	**48%**
Operation Mambo	Undirected	30	16	**46.66%**
Heroin Dealing	Undirected	37	15	**59.46%**
Montreal Street Gangs	Undirected	35	16	**54.28%**
Cocaine Dealers	Directed	28	23	**17.85%**
Operation Jake	Directed	38	29	**23.68%**
DrugNet	Directed	281	207	**26.33%**

Table 9. MICS Cardinalities for Drug Networks

Terror Networks	Network Type	Number of Nodes	MICS Cardinality	Reduction in Resources
Paris	Undirected	10	5	**50%**
Philippine	Undirected	14	6	**57.14%**
IS-E Zerkani	Undirected	39	17	**56.41%**
Madrid	Undirected	54	17	**68.52%**

Table 10. MICS Cardinalities for Terror Networks

had only a pair of nodes which formed “twins” (apart from the Philippine network, which two pairs of nodes which formed “twins”). Thus, the modified network had one less node (two in case of the Philippine network), as shown in Table 9 and Table 10. The East Turkestan network had multiple nodes which formed “twins” and a total of 7 nodes were condensed into a super node.

For the ease of understanding, we describe the results of the Operation Juanes drug network in detail. Out of 50 nodes in the network, if colors were injected in *only 22 nodes*, then all the nodes in the network would receive a unique identification. In other words, if authorities monitored just 22 individuals, then all the individuals in the network, could be uniquely identified. This indicates a *reduction* of $\frac{(50-22)}{50} \times 100$ = **56%** in resource requirements. Results for the other undirected drug networks are

Network	Domain	$\|V_1\|$	$\|V_2\|$	MDCS Cardinality	Reduction in Resources
MLB	Drug	12	83	11	**8.33%**
Noordin Top	Terror	4	15	3	**25%**
East Turkestan	Terror	20	64	15	**25%**

Table 11. MDCS Cardinalities for Drug and Terror Bipartite Graphs

presented in Table 9. The average reduction of resources is **52.92%** for the first five networks (undirected) and **22.62%** for the last three networks (directed). Similar results were obtained for the undirected terror networks. The Paris network is fairly small, hence the benefits of our approach aren't highlighted as much as the other networks. The average resource reduction, utilizing our approach, is **58.01%**.

Next, we analyze the bipartite network $G = (V_1 \cup V_2, E)$, where the node set V_1 represents the set of athletic trainers and health clinics suspected of providing steroid to the nodes in V_2, which represents the node set of MLB players. The cardinality of V_1 is 12 and that of V_2 is 83. Our objective is to monitor a subset of the 83 MLB athletes (by analyzing the drug test reports), so that all of the 12 drug providers would be uniquely monitored. Our analysis finds that if 11 athletes (out of 83) were monitored, then all 12 drug providers could have been exposed, resulting in a resource reduction of **8.33%**. The results for other bipartite graphs are presented in Table 11. The average reduction of resources utilizing the MDCS approach is **19.44%**.

3.3 Lower Bound for Fault Tolerant Identifying Codes

In prior discussions, we assumed that when a suspect becomes active in anomalous behavior, all of suspects' friends/associates will have some inkling of the intent of the suspect. Based on this assumption, we decided on the individuals to monitor so that if any one in the network is planning to behave anomalously, that individual can be uniquely identified. However, this assumption may be too strong and in reality may not always be true. If a signal (intent of anomalous behavior) does not reach the monitor (sensor monitoring a friend/associate) and as a consequence, the monitor cannot convey the information to the control center, the individual planning an attack cannot be uniquely identified. This scenario is equivalent to the scenario where the signal correctly reaches the monitor, but the monitor (due to some malfunction) fails to convey the information to the control center. Accordingly, it can be concluded that the system discussed previously does not have any fault-tolerant capability, in the sense that if a monitor (sensor) fails to convey any suspicious behavior to the control center, the individual planning to behave anomalously cannot be uniquely identified. However, the problem of inability to uniquely identify such an individual can be overcome by designing a more robust or fault-tolerant system. In this context, a system will be considered more *robust* if it can uniquely identify the individual, in spite of failure of one or more monitors to report any suspicious activity to the control center. An Identifying Code set that can tolerate up to k monitor failures will be referred to as $k-$ *Fault-tolerant (or Robust) Identifying Code.* In the following, we will establish a *lower bound* on the number of monitors that will be needed to design a $k-$ *Fault-tolerant Identifying Code* based system. It may be noted that although the authors in (Ray et al. 2004) also considered a robustness issues in Identifying Code context, the results presented in this chapter are completely different from the ones presented in (Ray et al. 2004).

3.3.1 Lower Bound on the Size of k Fault-tolerant Identifying Codes

Suppose that the graph contains N nodes. In order to be uniquely identifiable, each node must have a *unique signature/code* (or color/string) associated with it. With n bits, 2^n unique bit strings can be generated. However, one of these string comprises of all 0 bits, which cannot be the valid signature for a node, as a string comprising of all 0s also represents a scenario where no node is producing a signal. Accordingly, a lower bound on the size of Identifying Code for a N node system will be the smallest n such that $2^n \geq N + 1$ or $n \geq \lceil log\ (N + 1) \rceil$. Although with $n = \lceil log\ (N+1) \rceil$, unique codes for all N nodes can be generated, minimum *Hamming Distance* between a pair of codes in this case will be 1. However, with minimum code separation distance being equal to 1, it may be impossible to distinguish between two nodes, even when just one monitor is faulty. Consider two nodes v_1 and v_2 whose unique code/signature are the strings/colors A and AB. In this scenario, if the monitor B malfunctions, there will be no way for the control center to distinguish between the nodes v_1 and v_2. In order to be able to distinguish between two nodes when at most one monitor is faulty, the minimum code separation distance must be equal to 3. This is true as the minimum code separation distance being equal to 2 will also not be sufficient to distinguish between two nodes. Consider two nodes v_1 and v_2 whose unique code/signature are the strings/colors AB and AC. The Hamming distance between the codes is 2. However, in this case, if the control center receives a signal A, it will not be able to distinguish if its for node v_1 with monitor B being faulty or v_2 with monitor C being faulty. In order to design a $k-$ fault tolerant system, the minimum separation (Hamming) distance d between a pair of codes must be at least $2k + 1$, i.e., $d \geq 2k + 1$, or $k \leq \lfloor (d - 1)/2 \rfloor$.

Theorem 1. *A lower bound on the size of Identifying Code for a N node system that guarantees the minimum code separation distance of at least d is the smallest n_k, such that $n_k \geq \lceil log\ (N+1) \rceil + d - 1$.*

Proof. Suppose that to uniquely distinguish N nodes in a $k-$ fault tolerant system, each node must have a *unique signature/code* (or color/string) of n_k bits. Suppose that the bit string is represented by $b_{n_k-1}, b_{n_k-2}, \ldots, b_1, b_0$. With string length n_k, 2^{n_k} strings can be generated, but their minimum code separation distance will not be d (it will be 1). Two strings (codes) associated with two nodes will be at least distance d apart, only if at least in d positions of the corresponding strings, the bits are *inverse* of one another. One example of two such strings will be $b_{n_k-1}, b_{n_k-2}, \ldots, b_d, b_{d-1}, b_{d-2}, \ldots, b_1, b_0$ and $b_{n_k-1}, b_{n_k-2}, \ldots, b_d, \bar{b}_{d-1}, \bar{b}_{d-2}, \ldots, \bar{b}_1, \bar{b}_0$. By plugging-in any value (0 or 1), in the partial string $b_{n_k-1}, b_{n_k-2}, \ldots, b_d$ (partial string of length $n_k - 1 - d + 1 = n_k - d$), 2^{n_k-d} string can be generated. Each such string when concatenated with partial strings $b_{d-1}, b_{d-2}, \ldots, b_1, b_0$ and $\bar{b}_{d-1}, \bar{b}_{d-2}, \ldots, \bar{b}_1, \bar{b}_0$, will create $2^{n_k-d} \times 2 = 2^{n_k-d+1}$ strings of length n_k, whose minimum code separation distance will be d. Accordingly, a lower bound on the size of Identifying Code for a N node system that guarantees the minimum code separation distance at least d, will be the smallest n_k, such that $2^{n_k-d+1} \geq N+1$ or $n_k - d + 1 \geq \lceil log\ (N+1) \rceil$, or $n_k \geq \lceil log\ (N+1) \rceil + d - 1$, or $n_k \geq \lceil log\ (N+1) \rceil + 2k$. $\square$

3.4 Multiple Simultaneous Node Activations

Here, we consider the scenario where *two nodes simultaneously* become active in anomalous behavior. We present an Integer Linear Program (ILP) for determining the unique signatures for pairs of nodes. In this variant, not only individual nodes are required to have unique signatures, but also every pair of nodes are also required to have unique signatures. This additional requirement (with respect to the requirements seen thus far) necessitates that the ILP ensures that (i) a single node and a node pair do not end up having a identical signature, and (ii) two node pairs do not end up having a identical signature. Accordingly, the ILP is required to have four constraints: (i) coloring constraint (same as before), (ii) unique coloring 1-1 constraint (same as unique coloring constraint as before), (iii) unique coloring 1-2 constraint, and (iv) unique coloring 2-2 constraint. The 1-2 constraint ensures that the signature received by a single node cannot be the same as the signature received by a pair of nodes. In the following, we present the ILP for an undirected graph, as the ILPs for the directed and bipartite graphs will be similar to that of the ILP for the undirected graph.

Instance: $G = (V, E)$, an undirected graph.

Problem: Find the smallest subset $V' \subseteq V$, such that injection of colors at these nodes, ensures that each node v_i and each node pair $(v_j, v_k) \in V$, receive a unique color (either atomic or composite) through seepage.

We use the notation $N^+[v_i]$ to denote the closed neighborhood of v_i, for any $v_i \in V$. Corresponding to each $v_i \in V$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{v_i \in V} x_i$

Coloring Constraint: $\sum_{v_i \in N^+[v_j]} x_i \geq 1$, $\forall v_j \in V$

Unique Coloring 1-1 Constraint: $\sum_{v_i \in \{N^+[v_j] \bigoplus N^+[v_k]\}} x_i \geq 1$, $\forall v_j \neq v_k, \in V$

Unique Coloring 1-2 Const.: $\sum_{v_i \in \{N^+[v_j] \bigoplus (N^+[v_k] \cup N^+[v_l]\})} x_i \geq 1, \forall v_j \neq v_k \neq v_l, \in V$

Unique Coloring 2-2 Constraint:

$\sum_{v_i \in \{(N^+[v_j] \cup N^+[v_k]) \bigoplus (N^+[v_l] \cup N^+[v_m])\}} x_i \geq 1, \forall v_j \neq v_k \neq v_l \neq v_m, \in V$

The role of coloring constraint and unique coloring 1-1 constraint in this variant, is identical to the role of the coloring constraint and unique coloring constraint seen previously. These constraints ensure that no two nodes in the graph will have identical color. This variant, however, has two additional constraints.

Color assigned to a pair of nodes (v_k, v_l) is the union of the colors assigned to nodes v_k and v_l. The Unique Coloring 1-2 Constraint ensures that, for every pair of $(v_j, (v_k, v_l))$ in V, at least one node in the node set $N^+[v_j] \bigoplus (N^+[v_k] \cup N^+[v_l]) \subseteq V$ is injected with a color. This guarantees that v_j and (v_k, v_l) will not receive identical colors. In other words, this constraint ensures that for all combinations of distinct nodes $u, v, w \in V$, the node u will not have identical color as the node pair (v, w).

The Unique Coloring 2-2 Constraint ensures that, for every pair of $((v_j, v_k), (v_l, v_m))$ in V, at least one node in the node set $(N^+[v_j] \cup N^+[v_k]) \bigoplus (N^+[v_l] \cup N^+[v_m]) \subseteq V$ is injected with a color. This guarantees that (v_j, v_k) and (v_l, v_m) will not receive identical colors. In other words, this constraint ensures that for all combinations of distinct nodes $u, v, w, x \in V$, the node pair (u, v) will not have identical color as the node pair (w, x).

For the multiple activation scenario, we primarily focus on terrorist networks. We present the results of our approach when the simultaneous activation of two nodes is allowed. The ICS obtained for the Paris network, in this version, is $V' = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. In this network, there are 10 nodes (individuals) and $\binom{10}{2} =$

Network	Num. Nodes	Num. Colors Injected	Total Num. of Signatures Reqd.	Num. of Unique Signatures Produced
Paris	10	10	55	42
Philippines	14	14	105	37
Zerkani	39	39	780	692
Madrid	54	54	1485	1155

Table 12. Results for Multiple Simultaneous Activations

45 possible ordering of node pairs. Thus to have unique signatures for single nodes and node pairs, there must be a total of 10 + 45 = 55 unique signatures. Our analysis for this network shows that, even if colors are injected in all 10 nodes of the network, it creates only 42 unique signatures. This happens because five node pairs ((2, 6), (5, 6), (6, 7), (6, 9) and (6, 10)) receive the same color. Moreover, there are four instances where two node pairs ((1, 8), (4, 8)), ((2, 7), (2, 10)), ((3, 6), (6, 8)) and ((7, 8), (8, 10)) produce the same signature. These 13 non-unique signatures are a result of the topology of the network, i.e., the network is *not "twin-free"* for the multiple simultaneous activation scenario. This implies that, in the Paris network, further analysis is needed to uniquely identify the node pairs that becomes active. The results for the other three networks are tabulated in Table 12.

3.5 Superiority over Standard Graph Centrality Metrics

Note that, prior Drug and Terror related Social Network Analysis (SNA) studies have usually utilized network centrality measures to *identify key actors in the network* (Bright and Delaney 2013; Gialampoukidis et al. 2016; Berzinji, Kaati, and Rezine 2012). However, in this chapter, our objective is different in the sense that, we aim to *uniquely identify every actor in the network, should they become active in drug/terror related activities*, and not just the "key actors". Additionally, we show that if popular centrality metrics were utilized for realizing our objective, then the law enforcement agencies would end up wasting their resources. In the following, a brief description of the common social network centrality metrics is provided, before we present the results of our approach.

Centrality metrics have played a significant role in drug/terror SNA analysis (Bright and Delaney 2013; Gialampoukidis et al. 2016; Berzinji, Kaati, and Rezine 2012). These metrics have been primarily utilized in order to identify "key players/actors" in a network. However, as mentioned previously, the goal of this paper is not to identify only key players, but to uniquely identify all the players in a network. In other words, we claim that *all* the actors in the network are "key", in the sense that, all of them should be uniquely monitored. In order to show the efficacy of our Identifying Code based approach, we utilize the following standard centrality metrics for comparison namely, degree, betweenness, eigenvector, KPP-Neg (Borgatti 2006) and KPP-Pos (Borgatti 2006), as they have been extensively used in identifying key players in a network (Bright and Delaney 2013; Gialampoukidis et al. 2016; Berzinji, Kaati, and Rezine 2012; Borgatti 2006). In the following, we briefly describe the metrics that have been used in this chapter:

- Degree Centrality (DC): The Degree Centrality is the ratio of the number of neighbors of a node and the total number of edges in the graph/network (Freeman 1978). This fractional value lies in the interval $[0, 1]$. This metric is useful in the sense that, higher the DC value of a node, the more nodes it is connected to. In other words, a higher DC value of a particular node (player/actor) indicates that the player/actor is "important" since it is connected (friends) to (with) many nodes.
- Betweenness Centrality (BC): The Betweenness Centrality measures the extent

to which a node lies on paths between other nodes. Nodes with high betweenness may have considerable influence within a network by virtue of their control over information passing between others. They are also the ones whose removal from the network will most disrupt communications between other nodes because they lie on the largest number of paths taken by messages (Freeman 1978).

- Eigenvector Centrality (EV): Since some of the networks under study were directed networks, we decided to utilize the EC metric, as this is a natural extension of the DC metric. It may be noted that a node with a high in-degree does not necessarily have a high EC. Additionally, a node with high EC is not necessarily highly linked. In other words, a node is important if it is linked to by other important nodes (Bonacich 1972).
- Key Player Problem Negative (KPP-Neg): Borgatti in (Borgatti 2006) illustrated how standard centrality metrics described above failed in the identification of key players in the network. He defines KPP-Neg as the removal of the key player set that would result in a residual network with the least possible cohesion.
- Key Player Problem Positive (KPP-Pos): Borgatti in (Borgatti 2006) defined KPP-Pos as the key player set that is maximally connected to all other nodes.

We show that if law enforcement agencies utilized these five metrics to deploy agents in order to uniquely identify a player/actor in a network, as opposed to just identifying key players/actors, then they would end up wasting their resources (additional agents would be required to realize the objective), when compared to our Identifying Code based approach.

Now, we present the results of our analyses on undirected and directed drug/terror

Drug Networks	Network Type	Number of Nodes	MICS Cardinality	Reduction in Resources	DC MICS	Resource Wastage	BC MICS	Resource Wastage	EV MICS	Resource Wastage
Operation Juanes	Undirected	50	22	**56%**	45	**104.54%**	45	**104.54%**	45	**104.54%**
Operation Acero	Undirected	25	13	**48%**	23	**76.92%**	23	**76.92%**	23	**76.92%**
Operation Mambo	Undirected	30	16	**46.66%**	29	**81.25%**	29	**81.25%**	29	**81.25%**
Heroin Dealing	Undirected	37	15	**59.46%**	36	**140%**	36	**140%**	36	**140%**
Montreal Street Gangs	Undirected	35	16	**54.28%**	35	**118.75%**	35	**118.75%**	35	**118.75%**
Cocaine Dealers	Directed	28	23	**17.85%**	25	**8.69%**	28	**21.74%**	25	**8.69%**
Operation Jake	Directed	38	29	**23.68%**	36	**24.13%**	36	**24.13%**	36	**24.13%**
DrugNet	Directed	281	207	**26.33%**	281	**35.75%**	281	**35.75%**	281	**35.75%**

Table 13. MICS Cardinalities for Drug Networks

Terror Networks	Network Type	Number of Nodes	MICS Cardinality	Reduction in Resources	DC MICS	Resource Wastage	BC MICS	Resource Wastage	EV MICS	Resource Wastage
Paris	Undirected	10	5	**50%**	5	**0%**	5	**0%**	5	**0%**
Philippine	Undirected	14	6	**57.14%**	12	**100.00%**	12	**100.00%**	12	**100.00%**
IS-E Zerkani	Undirected	39	17	**56.41%**	33	**94.12%**	33	**94.12%**	33	**94.12%**
Madrid	Undirected	54	17	**68.52%**	51	**200.00%**	51	**200.00%**	51	**200.00%**

Table 14. MICS Cardinalities for Terror Networks

Drug/Terror Networks	MICS Cardinality	$KPPNeg$ MICS	Resource Wastage
Operation Juanes	22	48	**118.18%**
Operation Acero	13	22	**69.23%**
Operation Mambo	16	28	**75%**
Heroin Dealing	15	33	**120%**
Montreal Street Gangs	16	35	**118.75%**
Cocaine Dealers	23	28	**21.74%**
Operation Jake	29	37	**27.58%**
DrugNet	207	281	**35.75%**
Paris	5	5	**0%**
Philippine	6	12	**100%**
IS-E Zerkani	17	33	**94.12%**
Madrid	17	49	**188.23%**

Table 15. Comparison of MICS with KPP-Neg

networks. Recall that, some of the networks under study were not "twin-free". For the ease of understanding, we describe the results of the Operation Juanes drug network in detail. Out of 50 nodes in the network, if colors were injected in *only 22 nodes*, then all the nodes in the network would receive a unique identification. In other words, if authorities monitored just 22 individuals, then all the individuals in the network, could be uniquely identified. This indicates a *reduction* of $\frac{(50-22)}{50} \times 100 =$ **56%** in

Drug/Terror Networks	MICS Cardinality	Number of Nodes Uniquely Monitored	$KPPPos$ K Values	Number of Nodes Uniquely Monitored	% of Individuals Uniquely Monitored
Operation Juanes	22	50	22	38	**38/50 = 76%**
Operation Acero	13	25	13	15	**15/25 = 60%**
Operation Mambo	16	30	16	20	**20/30 = 66.66%**
Heroin Dealing	15	37	15	29	**29/37 = 78.37%**
Montreal Street Gangs	16	35	16	21	**21/35 = 60%**
Cocaine Dealers	23	28	23	22	**22/28 = 78.57%**
Operation Jake	29	38	29	34	**34/38 = 89.47%**
DrugNet	207	281	207	213	**213/281 = 75.80%**
Paris	5	10	5	10	**10/10 = 100%**
Philippine	6	14	6	7	**7/14 = 50%**
IS-E Zerkani	17	39	17	25	**25/39 = 64.10%**
Madrid	17	54	17	37	**37/54 = 68.52%**

Table 16. Comparison of MICS with KPP-Pos

resource requirements. Furthermore, we compare the MICS cardinality following the Identifying Code approach with the MICS cardinalities obtained by following the standard SNA centrality approaches - degree centrality (DC), betweenness centrality (BC) and eigen vector centrality (EV). We observe that if we were to adopt these standard SNA metrics as monitoring strategies, then law enforcement agents will need to monitor 45 individuals with the *highest centrality scores*, to ensure that each node in the network has unique identification. Individuals with the highest centrality scores are an indication of the importance of the individual to the respective drug network (Bright, Hughes, and Chalmers 2012; Bright and Delaney 2013; Bright et al. 2015). Requiring 45 nodes instead of the MICS solution of 22, clearly results in wastage of resources. For this network, the wastage amounts to $\frac{(45-22)}{22} \times 100 =$ **104.54%**. This is somewhat *counter-intuitive* in the sense that, authorities would waste resources if they were to *monitor only important individuals.* Identifying Code saves resources as it takes the entire network (a global view) into account rather than the individual importance of respective nodes (a local view). In other words, for optimal resource allocation for unique monitoring of individuals associated with

DTOs/TOs, the law enforcement authorities *must* consider a combination of important as well as unimportant individuals.

Results for the other undirected drug networks are presented in Table 13. The average reduction of resources is **52.92%** for the first five networks (undirected) and **22.62%** for the last three networks (directed). The average resource wastage, by following the DC strategy, for both undirected and directed drug graphs are **104.29%** and **22.86%** respectively. Finally, the average resource wastage for undirected and directed drug graphs, by following the BC and EV strategies, are **104.29%**, **27.21%** and **104.29%**, **22.86%**, respectively.

Similar results were obtained for the undirected terror networks and is presented in Table 14. The Paris network is fairly small, hence the benefits of our approach aren't highlighted as much as the other networks. The average resource reduction, utilizing our approach, is **58.01%**. The average resource wastage for the undirected terror graphs, by following the three common centrality strategies, is **98.53%**.

Having shown the superiority of the Identifying Codes approach over the standard centrality metrics, we next compared our approach with the metrics provided by (Borgatti 2006). This is because, (Borgatti 2006) showed how his metrics, KKP-Neg and KPP-Pos, are superior than the standard centrality metrics when monitoring "key" players in a network. In Table 15, we compare the cardinality of the MICS based approach with the cardinality of the KPP-Neg based approach. In other words, for the 50 individual Operation Juanes network, simply monitoring 22 individuals results in unique monitoring of all the individuals in the network. However, following the KPP-Neg metric, the law enforcement agencies must monitor 48 individuals in order to ensure that all the nodes in the network are uniquely monitored. This results in a staggering **118.18%** wastage in resources. The results for the other drug/terror networks are presented in Table 15. The average resource wastage of the KPP-Neg metric for undirected drug and terror networks is **98.16%** and that for directed drug networks is **28.36%**.

For the KPP-Pos metric, we set the value of K to be equal to the cardinality of the MICS solution for the corresponding network, and proceeded to compute the percentage of the nodes which would be uniquely monitored, by assigning K law enforcement agents to the network. It is trivial to note that for the MICS solution, 100% of the individuals in the network will be uniquely monitored. In Table 16, for the Operation Juanes network, monitoring $K = 22$ nodes results in the network being **76%** uniquely monitored by following the KPP-Pos metric. This implies that by following the KPP-Pos metric, we are unable to uniquely monitor the *entire* network. The results for the other drug/terror networks are presented in Table 16. The average % of the individuals uniquely monitored following the KPP-Pos metric for undirected drug and terror networks is **69.29%** and that for directed drug networks is **81.28%**.

3.6 Augmented Identifying Codes (AIC)

In this variant of the Identifying Code problem, we have assumed that there are already certain law enforcement authorities assigned to monitor certain individuals in the respective drug or terror networks. We have further assumed that this assignment does not uniquely identify all the individuals in the network. In other words, following the Identifying Code strategy, law enforcement agencies had the *complete freedom* to deploy agents to ensure that *all the individuals in the network were uniquely monitored*, in a sense, a clean slate approach. The AIC strategy restricts this freedom in the sense that, there are already a certain number of agents deployed to monitor certain individuals, but additional agents are required to ensure *unique monitoring* of *all* individuals in the network. This is a more *realistic* version in the sense that, as law enforcement agencies have already deployed agents in the field to monitor individuals associated with DTOs and TOs. For undirected and directed graphs, the general objective of the AIC problem is to determine the smallest set of individuals to monitor, V_2, *in addition to the individuals currently being monitored*, V_1, in order to ensure that each node in V can be uniquely monitored.

Instance: $G = (V, E)$, an undirected graph and a set $V_1 \subseteq V$, the set of nodes currently being monitored.

Problem: Find the smallest subset $V_2 \subseteq V$, such that $N^+[v] \cap V_3$ is unique for each node $v \in V$ (each node receives a unique color, either atomic/composite, through seepage), and $V_3 = V_1 \cup V_2$.

We use the notation $N^+[v_i]$ to denote the closed neighborhood of v_i, $\forall v_i \in V$. Corresponding to each $v_i \in V$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{v_i \in V} x_i$

Coloring Constraint: $\sum_{v_i \in N^+[v_j]} x_i \geq 1$, $\forall v_j \in V$

Unique Coloring Constraint:

$$\sum_{v_i \in \{N^+[v_j] \bigoplus N^+[v_k]\}} x_i \geq 1,\ \forall v_j \neq v_k, \in V$$

Initial Set Constraint: $x_i = 1, \forall v_i \in V_1$

Network	Num. Nodes	MICS	Centrality Measure											
			Degree				Betweenness				Eigen Vector			
			$k = 25\%$	$k = 50\%$	$k = 75\%$	$k = 100\%$	$k = 25\%$	$k = 50\%$	$k = 75\%$	$k = 100\%$	$k = 25\%$	$k = 50\%$	$k = 75\%$	$k = 100\%$
Op. Juanes	50	22	12 + 15	25 + 9.5	38 + 8.77	45 + 0	12 + 15	25 + 9.5	38 + 8.77	45 + 0	12 + 15	25 + 9.5	38 + 8.77	45 + 0
Op. Acero	25	13	6 + 9	12 + 6	18 + 5	23 + 0	6 + 9	12 + 6	18 + 5	23 + 0	6 + 9	12 + 4	18 + 5	23 + 0
Op. Mambo	30	16	7 + 11	15 + 7.5	22 + 6.33	29 + 0	7 + 11	15 + 7.5	22 + 6.33	29 + 0	7 + 11	15 + 7.5	22 + 6.33	29 + 0
H. Dealing	37	15	9 + 10	18 + 8.6	27 + 4.5	36 + 0	9 + 10	18 + 8.6	27 + 4.5	36 + 0	9 + 10	18 + 8.6	27 + 4.5	36 + 0
M. Gangs	35	16	9 + 9	18 + 8.8	27 + 6.28	35 + 0	9 + 9	18 + 8.8	27 + 6.28	35 + 0	9 + 9	18 + 8.8	27 + 6.28	35 + 0
C. Dealers	28	23	7 + 20	14 + 14	21 + 7	25 + 0	7 + 20	14 + 14	21 + 7	25 + 0	7 + 20	14 + 14	21 + 7	25 + 0
Op. Jake	38	29	9 + 22	20 + 15.16	29 + 7.2	36 + 0	9 + 22	20 + 15.16	29 + 7.2	36 + 0	9 + 22	20 + 15.16	29 + 7.2	36 + 0
Drugnet	281	207	70 + 162.48	140 + 131.76	210 + 71	281 + 0	70 + 162.47	140 + 131.76	210 + 71	281 + 0	70 + 162.47	140 + 131.76	210 + 71	281 + 0

Table 17. Augmented MICS Cardinalities for Drug Networks Corresponding to Various Centrality Measures

Network	Num. Nodes	MICS	Centrality Measure											
			Degree				Betweenness				Eigen Vector			
			$k = 25\%$	$k = 50\%$	$k = 75\%$	$k = 100\%$	$k = 25\%$	$k = 50\%$	$k = 75\%$	$k = 100\%$	$k = 25\%$	$k = 50\%$	$k = 75\%$	$k = 100\%$
Paris	10	5	2 + 4	5 + 1.67	7 + 0	5 + 0	2 + 4	5 + 1.67	7 + 0	5 + 0	2 + 4	5 + 1.67	7 + 0	5 + 0
Philippine	14	6	3 + 11	7 + 7	10 + 4	12 + 0	3 + 11	7 + 7	10 + 4	12 + 0	3 + 11	7 + 7	10 + 4	12 + 0
IS-E Zerkani	39	17	10 + 15	20 + 10	30 + 7.5	33 + 0	10 + 15	20 + 10	30 + 7.5	33 + 0	10 + 15	20 + 10	30 + 7.5	33 + 0
Madrid	54	17	13 + 23	27 + 17	40 + 13	51 + 0	13 + 23	27 + 17	40 + 13	51 + 0	13 + 23	27 + 17	40 + 13	51 + 0

Table 18. Augmented MICS Cardinalities for Terror Networks Corresponding to Various Centrality Measures

For simulation purposes, we assumed that the Initial Set of nodes being currently monitored are the $k-$nodes with the highest degree, betweenness and eigen vector centrality scores, i.e., the k most important nodes (individuals) in the network. For each drug/terror network, we present the number of nodes in the network, the MICS cardinality of the network and the cardinality of the AIC set, if $k-$nodes were currently monitored (where k is varied from 25%, 50%, 75% and 100% of the total nodes in the network), for each centrality metric. It may be noted that some nodes in Initial Set may have equal centrality scores with the nodes not present in the Initial Set. For example, consider the following centrality score sequence of 6 nodes, ordered descendingly: 0.88, 0.88, 0.88, 0.85, 0.77, 0.77. The top 25% of these nodes would comprise of nodes 1 and 2 which have 0.88 as their centrality scores. It can be seen that node 3 also has a score of 0.88. The AIC solution cardinality obtained by initially monitoring nodes 1 and 2 may be different from the AIC solution cardinalities obtained by monitoring nodes 1 and 3, and nodes 2 and 3 respectively. As a result, all such combinations should be considered and the AIC solution cardinality should be averaged over the number of combinations considered. It should be noted that for the

$k = 100\%$ scenario, we have assumed that authorities can deploy *at most* n number of agents, for the unique monitoring of n individuals in the network. The Operation Juanes drug network has 50 nodes in the network and the MICS cardinality of such a network is 22. For $k = 25\%$ of degree centrality, we considered the top 25% nodes with the highest degree centrality score which turns out to be 12. Furthermore, an additional 15 nodes have to be monitored to ensure that every node in the network has a unique identification, resulting in an AIC cardinality of 27. As all 12 nodes in the Initial Set had unique degree centrality scores when compared to the nodes not in the Initial Set, only 15 additional individuals have to be monitored for unique identification of all the nodes in the network. For $k = 50\%$ we observe that some of the 25 nodes in the Initial Set had the same degree centrality with nodes not in the Initial Set. As described previously, all such combinations were generated and the average cardinality of the additional set of monitors required turns out to be 9.5, resulting in the average AIC cardinality of 34.5. Similarly, the cardinality of the additional set of monitors for $k = 75\%$ turns out to be 8.77, resulting in the average AIC cardinality of 46.77. For $k = 100\%$, we observe that the law enforcement agencies will need to monitor 45 individuals with the highest centrality scores, to ensure that each node in the network has unique identification.The results for other centrality metrics of other drug networks are shown in Table 17.

It can be observed that the AIC cardinality is greater than the MICS cardinality for all the drug networks. Ideally, in the Operation Juanes network, for $k = 25\%, 50\%, 75\%$ and 100%, *if the nodes in the Initial Set were a subset of the MICS*, then the authorities would only need to monitor an additional 10, 0, 0 and 0 individuals respectively, to get unique identifications for all the nodes in the network. The fact that the authorities require an additional 15, 9.5 and 8.77 individuals (on average), indicate that there was some sub-optimal initial monitoring of individuals (Initial Set), essentially leading to a wastage in resources. Only for the $k = 100\%$ scenario (when they have complete freedom of deploying agents) do the agencies not require any additional resources, but such a scenario directly corresponds to the MICS strategy, where we have shown that far lesser resources are required for unique monitoring of individuals associated with DTOs/TOs. Finally, similar to our MICS approach, these results are counter-intuitive in the sense that the Initial Set consists of the top k important individuals. In other words, if the law enforcement authorities were to monitor *only* important individuals, then they would require additional resources, for unique monitoring of the *entire* network of individuals. Hence, it can be said that our approach provides an optimal balance between monitoring important as well as unimportant individuals. The AIC results for the TOs are presented in Table 18.

3.7 Discussions

Here, we provide some insights into scenarios where there may be fundamental issues with the underlying datasets. Broadly speaking, there may be two such scenarios:

- Omission Errors: In this scenario, the entire drug or terror network has not been completely identified. In other words, not all actors and/or the relationships between actors are known. If such a scenario were to arise, then the law enforcement agency can do the following. For the network that is known (a sub graph of the actual graph), the agency can utilize our Identifying Code based approach and optimally monitor that network. However, as additional actors and relationships are discovered by agents and other sources, the agency can utilize our AIC based approach to deploy additional sensors, in order to uniquely monitor the newly identified actors.

- Commission Errors: In this scenario, actors and relationships that have been reported, are *not actually present in reality.* In such errors, the law enforcement agency can either, (i) decide to deploy agents on actors who are not present (inject colors to nodes which do not exist) or, (ii) decide to monitor the neighbor(s) of the actors who are not present (inject colors to the nodes which exist, and are adjacent to nodes which have been falsely reported to exist). In the former, if the node on which a color is to be injected (place an agent on) does not exist, then the resulting graph (after the removal of the node and edges incident to it), will have twin vertices (nodes), and hence, two or more nodes may end up with the same color. In such cases, we have to condense the twin vertices (nodes), as explained earlier, and the agency can utilize the Identifying Code based strategy to effectively monitor the network. In the latter, it is trivial to note that if a node other than a coloring node (node on which color is injected) is removed from the network, then we do not end up with "twins" in the network, and each and every node in the network retains its unique identification, after the removal of the non-coloring node. Hence, the agency does not have to do any additional computation.

3.8 Future Research Directions

All of the works presented as a part of this chapter all aim to disrupt the operations of criminal organizations by identifying individuals who engage in anomalous behaviors.

That being said, however, there are multiple research avenues which could be examined in this domain. To highlight a few,

- *Generalize Multiple Simultaneous Activations*: In this chapter, we introduced a novel concept where two nodes could simultaneously become active in anomalous behavior. For our approach to be applicable to the real world setting, it would be interesting to develop algorithms for the scenario where k multiple simultaneously activations can be effectively handled. In other words, this implies a need for the development of a generalized approach to detect multiple simultaneous activations as opposed to the much more restrictive approach presented in this chapter.
- *Probabilistic Identifying Codes*: One key assumption made in our problem setup is that, when an individual (or a node) engages in anomalous behavior, all of its friends/associates (neighbors) become aware of it. Generally speaking, this is a purely deterministic setting and may not be a valid assumption in the real world. A probabilistic setting may be even more appropriate, where there is a probability value associated with each edge in the network. These probability values would indicate the probabilities of the signal (anomalous behavior) reaching the neighbors of the node engaging in anomalous behavior.
- *Game Theoretic Monitoring Strategies*: This problem could be setup as a two player attacker-defender game where the attackers are the suspect individuals in the network and the defender is the network operator. The MICS problem discussed thus far is a static placement of sensors. As a result, the attackers could learn the placement of these sensors and modify their behaviors accordingly. Game theory can be introduced to determine a moving placement strategy which would ensure unique identification of the suspect individuals, thereby making it

even more difficult for the attackers to evade capture when they become engaged in anomalous behavior.

Chapter 4

CRITICAL INFRASTRUCTURE NETWORKS

The electric power grid is arguably the most critical of all the infrastructures as other infrastructures, such as, communication, transportation and finance are heavily dependent on it. Similarly, High Voltage (HV) power transformers, generators, and transmission lines are the most critical components of the electric power grid. The large generators are located inside enclosed structures, and are constantly (and often redundantly) monitored by a multitude of dedicated instruments. On the other hand, the power transformers are typically located in open-air switch-yards, where they are at the mercy of nature's elements as well as (more recently) trigger-happy humans (Wikipedia, n.d.). Therefore, an untimely loss of HV transformers can be catastrophic for not only the electrical infrastructure, but also the other critical infrastructures that depend on it. Accordingly, it will be helpful if it can be recognized before the event, that a transformer is heading towards a failure, so that corrective measures can be undertaken. Fortunately, before a transformer reaches its critical failure state, there are "cues"(or indicators) which, if monitored periodically, can alert an operator that the transformer is heading towards a failure. One of the indicators is the Signal to Noise Ratio (SNR) of the voltage and current signals in substations located in the vicinity of the transformer. During normal operations, the width of the SNR bands are small. However, when the transformer heads towards a failure, the widths of the bands increase, reaching their maximum just before the failure actually occurs. This change in width of the SNR can be observed by Phasor Measurement Units (PMUs) located nearby.

PMU is a device that can be utilized as a "sensor" for monitoring the health of transformers. When placed on a generator, load, or zero injection bus, in the power grid, PMUs give the voltage of that particular bus, as well as the currents flowing in the branches (lines or transformers) incident on that bus (while being subjected to the PMU's measurement channel limitations). Since a power transformer can *only* be placed between two buses, a judicious placement of a few PMUs (sensors) can effectively monitor health of all the transformers, and in case of a transformer heading towards a failure, the sensors can create a *unique fault signature* that enables the operator to identify the troubled transformer.

It may be noted that a variety of devices such as (a) Buchholz relay, (b) Pressure relay, (c) Oil level monitor device, (d) Winding thermometer, (e) Differential relay, (f) Overcurrent relay, and (g) Ground current relay are available to monitor the health of transformers. However, these devices are unable to provide *unique* identification of a failing transformer, as was indicated by the Rudd power transformer failure incident (News 2016). Devices (a)-(d) monitor the mechanical properties, while (e)-(g) oversee the electrical properties of the transformer. The Buchholz relay detects mechanical faults that are induced by electrical faults in oil-immersed transformers. The pressure relay detects sudden rate-of-increase of pressure inside the tap changer oil enclosure. The oil level monitor and winding thermometer detect increased temperatures inside the oil and windings of the transformer, respectively. The differential relay checks for faults inside the transformer by comparing the currents flowing through its primary and secondary windings. The overcurrent and ground current relays look for abnormal currents in the transformer under special conditions (in-rush currents during energizing, ground currents during faults, etc.). However, incidents such as the Rudd power transformer failure indicate that these devices alone may not be always effective in monitoring a power transformer's health.

The problem of placing these sensors, for monitoring the power grid, has been studied by multiple researchers over the past decade (Salehi et al. 2012; Pal, Vullikanti, and Ravi 2016). Additionally, with the continuous discovery of real-world attacks such as Stuxnet (Karnouskos 2011), Dragonfly (Team 2017) and a wide range of cyberattacks– jamming, Denial of Service, packet dropping, false-data injection and compromise of data integrity (Nandanoori et al. 2020; Niu and Clark 2019)– robustness of existing sensor placement mechanisms becomes critical. Thus, in this chapter, we leverage the ideas of Minimum Discriminating Code Set (MDCS) based PMU placement (Basu et al. 2019; Basu, Padhee, et al. 2018) and Moving Target Defense (MTD) in cybersecurity (Jajodia et al. 2011; Sengupta et al. 2017) to build a defense-in-depth solution.

Such solutions continuously move the detection surface to make it challenging for an adversary to impede the unique identification of failure signals of HVTs. While PMUs are difficult to move, as opposed to the movement of physical resources in security games (Paruchuri et al. 2008), once placed, they can be efficiently activated and deactivated, similar to the dynamic movement in intrusion detection systems (Sengupta et al. 2018). While one may choose to activate all the PMUs placed upfront, the cost of maintaining them can become an impediment. Hence, the periodic use of a smaller subset (that still ensures unique identification) of the sensors placed upfront can be considered. Further, work in MTD has relied solely on heuristic guidance when constructing the configuration set that can result in all defenses being vulnerable to one attack, i.e. it is *not differentially immune* (Sengupta, Chakraborti, and Kambhampati 2019).

Identifying Code is a mathematical tool that enables one to *uniquely* identify one or more *objects of interest*, by generating a *unique signature* corresponding to those objects, which can then be detected by a sensor. In this chapter, the objects of interest are HV transformers. When a transformer is heading towards failure, it generates "indicators", which, if monitored by some "sensors", may provide information to an operator in the control center about the impending failure of the transformer. Since the number of transformers in the grid is large, and the sensors are expensive, one would like to deploy as few sensors as possible (fewer than the number of transformers) and yet retain the capability that, when a transformer is heading towards a failure, it can be *uniquely* identified.

In this chapter, we first describe how Identifying Codes and its special variant, called the Discriminating Codes, can be utilized for *unique* identification of the transformers that are heading towards a failure. We show how a power grid can be modeled as a graph and how entities of interest (HV Transformers in this case) can be modeled as a bipartite graph. Next, we provide a solution technique to solve the problem of uniquely monitoring the transformers. In the second half of the chapter, we focus primarily on the adversarial scenario. First, we define a novel variant of the problem discussed in the first part of the chapter, called the $K-$differentially Immune MDCS (hereafter K-δMDCS). We find K MDCSs of a graph, in which all K solutions can uniquely identify failing HVTs, with the added constraint that no two MDCSs share a common vertex; thus resulting in a differentially immune configuration set for the MTD. Given that the original MDCS problem is NP-Complete, we show that K-δMDCS is also NP-Complete and provide an optimal Quadratically Constrained Integer Linear Programming (QC-ILP) approach to find the $K_{\max}$-MDCS of a graph. While our approach proves scalable for large power networks (MATPOWER IEEE test cases), we also propose a greedy approach that is computationally faster but trades-off on finding the largest K value. Second, we model the interaction between the power utility company (hereafter, the defender) and the adversary, as a normal-form game. The notion of Strong Stackelberg equilibrium used in this game-theoretic formulation, popular in existing literature (Sinha et al. 2015; Sengupta et al. 2017), assumes a strong-threat model and aids in finding a good sensor activation strategy for the defender. Finally, we conclude the chapter with a brief discussion about future research directions.

4.1 Related Work

With regards to PMUs, research on maintenance has primarily been restricted to the reliability of the PMU device itself. (Murthy et al. 2014) used fuzzy logic to perform reliability assessment of PMUs. In (Becejac, Dehghanian, and Kezunovic 2016), the authors analyzed how maintenance schedule of the PMU device itself can be updated based on its outputs. An example depicting the use of SNR in predicting actual equipment failure was presented in (Jones, Pal, and Thorp 2014). However, the equipment whose failure was demonstrated in (Jones, Pal, and Thorp 2014) was a single-phase potential transformer, and not three-phase power transformers which are the focus of the study here. Although researchers have analyzed causes of blackouts and outages from an *operational failure* perspective, not much significance has been attributed to the role that critical equipments can play in alleviating or exacerbating a given contingency. The failure of a vital equipment at a crucial time may considerably worsen system stress. Therefore, real-time health monitoring of critical power system assets is a task worth undertaking.

Prior research on health monitoring using PMUs have been mostly directed towards improving security and stability of the power system (Salehi et al. 2012). In addition, a number of studies have focused on placement of PMUs (Pal, Vullikanti, and Ravi 2016; Pal et al. 2017) to realize a variety of objectives. The problems under study in this chapter can also be viewed as a PMU placement problem as it computes the fewest number of PMUs and their locations, so that the *unique* identification capability is realized. It is important to highlight here that none of the PMU placement strategies proposed so far had the unique identification capability as the objective for PMU deployment.

Adversarial attacks on power grids comprise of false-data injection, jamming, DoS and packet-dropping attacks (Deka, Baldick, and Vishwanath 2015; Deng et al. 2016; Nandanoori et al. 2020). While researchers have proposed a multitude of defense mechanisms (Tan et al. 2017), including Moving Target Defense (MTDs) (Chatfield and Haddad, n.d.; Potteiger et al. 2020), they do not consider the problem of sensor placement to monitor HVTs. On the other hand, works that leverage the formalism of Discriminating Code Sets (Charbit et al. 2006) to optimize sensor placement (Basu, Padhee, et al. 2018), have focused on scalability issues and provided theoretical bounds in these settings (Basu et al. 2019); completely ignoring the issue of robustness to adversarial intent.

While an array of research work has formally investigated the notion of finding an optimal movement function M for MTDs, the configuration set C is pre-decided based on heuristic guidance from security experts (Sengupta, Chowdhary, Sabur, et al. 2020). While some works consider the aspect of differential immunity by analyzing code overlap for cyber systems (Carter, Riordan, and Okhravi 2014) or Jacobians of gradients for deep neural networks (Adam et al. 2018), these measures have no way

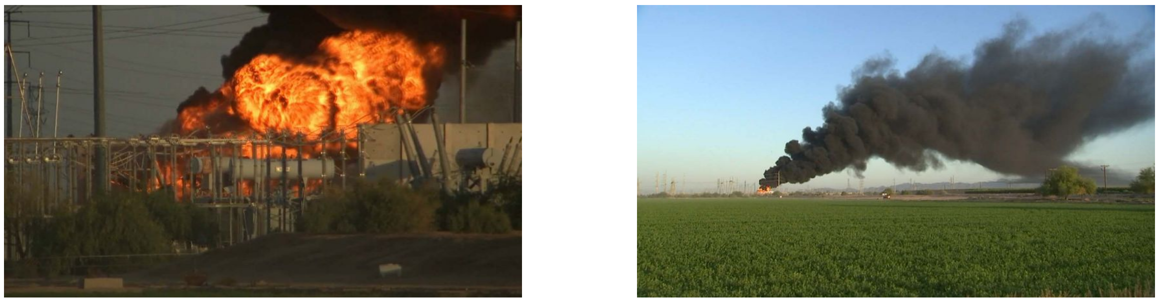


Figure 8. Transformer fire at Salt River Project's Rudd substation in Avondale, AZ

of ensuring differential immunity. The notion of k-set diverse solutions in Constraint Satisfaction Programming (CSP) (Hebrard et al. 2005), although conceptually similar to our notion of differential immunity, does not have the added constraint of finding a minimum sized solution (as in the case of MDCS). In adversarial scenarios, our work is the first to formalize the notion of diversity in graphs and propose linear programming methods to find them.

4.2 Monitoring the Health of Critical Power System Equipments

During the early hours of June 1, 2016, a large power transformer at the Rudd substation of Salt River Project (SRP), a large utility company in Arizona, suddenly caught fire. A 27,000-gallon tank of mineral oil used as a transformer coolant, burned, and spewed thick smoke over a large area. A few snapshots are illustrated in Figure 8 (News 2016). The cause of the failure was identified to be bushing failure. Due to the redundancy present in the system design as well as the fact that the fire broke out during low-load conditions (system load is small in early morning), no power outages occurred. This incident highlights the need for better monitoring techniques for these critical and expensive equipments.

SRP shared their operational data leading up to the failure of this transformer with us for analysis. Since causes of such failures gradually build-up over time, if one is paying attention, the signs of an impending failure may be observable “days” before the actual failure event. PMUs continuously produce outputs at a very fast rate (typically 30 samples per second). When placed near transformers, PMUs, through their measurements, can serve as *sensors* to monitor the health of the transformer, and capture degradation in the health of a transformer over time. It may be noted that a PMU provides complex voltage and current measurements at the bus where it is placed. If the PMU has to serve as a sensor for monitoring transformer health, it must have a way to measure it with a “cue” (or indicator or metric). This metric should be independent of the “unit” of the measured quantity (either voltage or current), so that a proper comparison can be made. Signal to noise ratio (SNR), a classical measure of the quality of a signal, can serve as this desired metric. It compares the level of a signal to the level of background noise that is present in it. Mathematically, the SNR of a signal can be expressed as a reciprocal of the coefficient of variation, i.e., the ratio of its mean to its standard deviation, as shown in Equation 4.1.

$$SNR\ (in\ dB) = 10 * log\frac{\mu}{\sigma} \tag{4.1}$$

In Equation 4.1, μ is the signal mean or expected value and σ is the standard deviation, or an estimate thereof. It is difficult to directly compare different signals (such as voltages and currents). However, SNR (in decibels) is a relative metric and therefore, it can be used to compare diverse signals and create alerts/alarms. The Rudd transformer failure data obtained from SRP, comprises of PMU readings (voltages and currents) one year away from the day of the failure (June 1, 2016) up to the data collected only a few hours prior to the actual failure event.

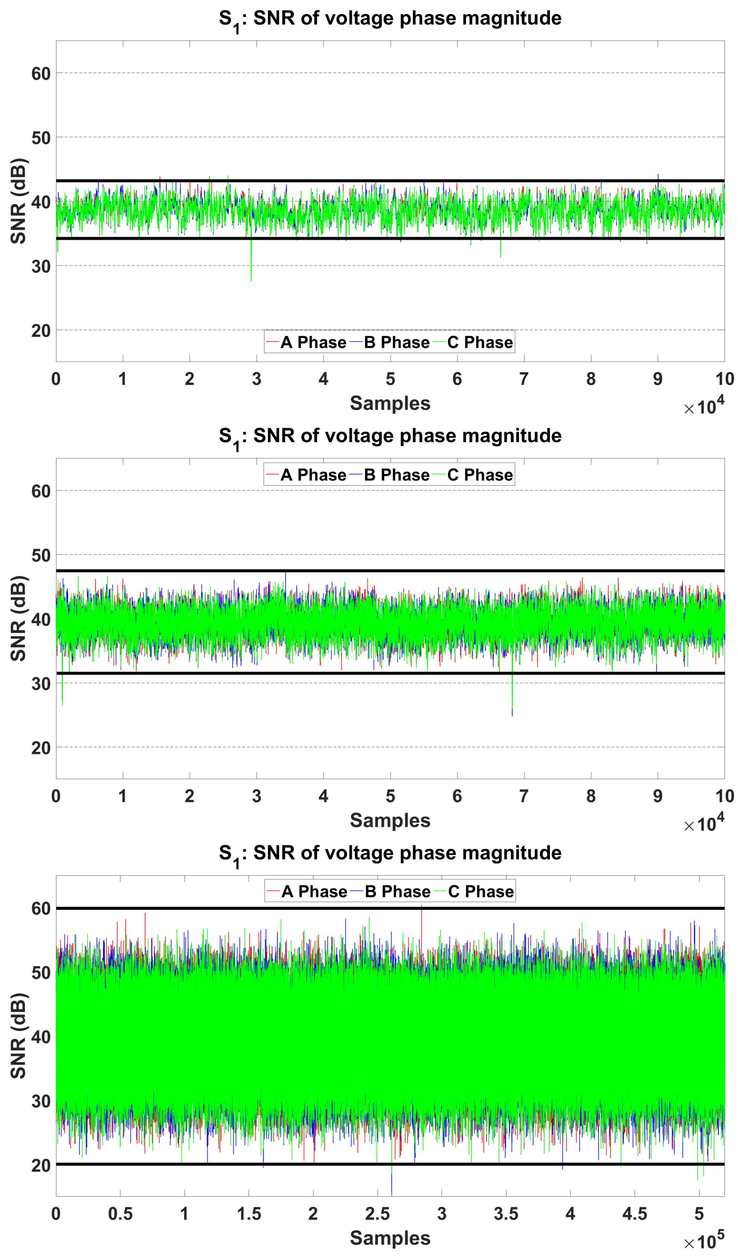


Figure 9. Variation in width of SNR as one moves closer (in time) to instant of failure.

Two important pieces of observation were made from the SRP data.

Observation 1: A steady growth in the width of the SNR bands (computed from the voltage magnitude measurements obtained from neighboring substations), was observed over a period of time, till the transformer failed. The observations for three

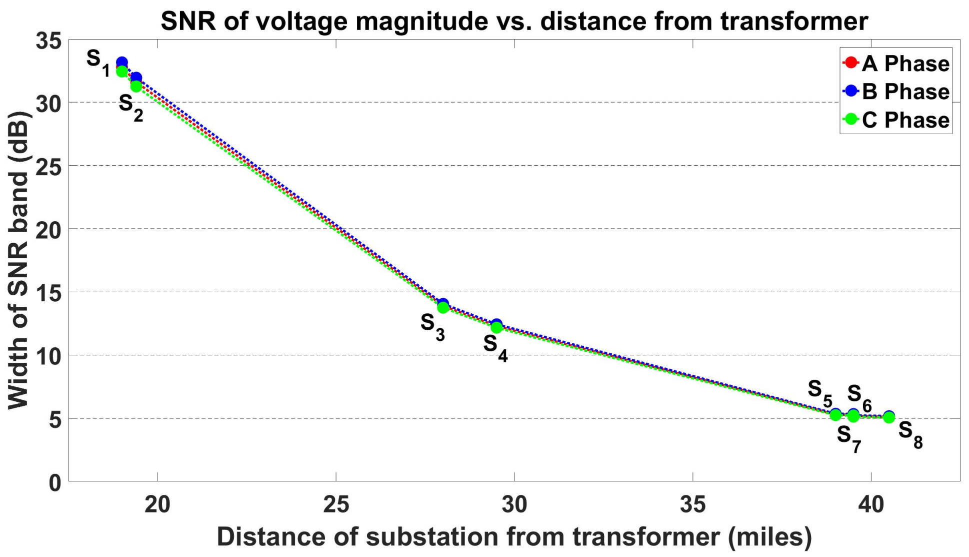


Figure 10. Standard deviation of width of SNR as one moves (spatially) away from the failing equipment

instances of time, as it approached the actual time of failure, are shown in Figure 9. Since the growth was similar in all three phases, it was concluded that the SNRs were capturing an event that was affecting all three phases, and not due to a single phase failure event, contributed by a current or a potential transformer failure. Moreover, as the width was uniform over the observed time period (an hour worth of data), it is clear that the captured event was *not* a random transient event.

Observation 2: In observation 1, we noted that the width of the SNR band at a specific PMU (sensor) location, increases as time approaches the actual failure event. From the data it was also clear that, as the distance of the PMU (sensor/monitoring device), from the transformer (monitored device) increased, the width of the observed SNR decreased. Figure 10 shows the decrease in the width of the SNR bands as a function of the electrical distance (termed as hops) from the Rudd transformer. The data was collected from eight substations (S1, ..., S8) that neighbor Rudd, and had

PMUs placed on them. It may be noted that the Rudd substation itself did not have a PMU on it during the time of failure.

Given that the deteriorating condition of a transformer can be noticed by PMUs located within a certain distance of the transformer, signals indicating the deteriorating condition, can be utilized to deploy effective monitoring strategies, so that an alarm is generated *before* a transformer reaches a critical failure state. *Identifying Code* is a mathematical tool that can be used for monitoring transformers in the power grid. Using this technique, the fewest number of sensors needed to enable an operator to *uniquely* identify the failing transformer before it reaches a critical failure state can be computed.

Identifying Code is useful when the goal is to monitor all nodes of the graph (i.e., each node is required to have a unique signature). However, in this chapter our focus is on monitoring the health of only power transformers. Moreover, in Identifying Codes, a color can be injected at any node of the graph (i.e., a sensor can be placed at any node of the graph). However, in the health monitoring problem, a sensor placed far away from the equipment to be monitored, may not be useful as "cues" (signals) indicating failing state of the equipment, may not even reach this sensor because of its distance from the equipment. Accordingly, some modification to the original concept of *Identification* is needed. The following modifications are sufficient to capture the new scenario: (i) We identify a subset $V' \subseteq V$ that needs to receive a unique color; (ii) For each node $v \in V'$, we compute $N^k(v)$, where $N^k(v)$ represents the k-hop neighbors of v (i.e., the set of nodes in the graph whose shortest path distance to v is at most k); (iii). We construct a Bipartite graph $G' = (V_1 \cup V_2, E)$ such that (a) $V_1 = V'$, (b) $V_2 = \cup_{v \in V_1} N^k(v)$, and (iii) for nodes $v_i \in V_1$ and $v_j \in V_2$, there is an edge $e \in E$, if and only if $v_j \in N^k(v_i)$. With this modification, the transformer health monitoring problem with the fewest number of sensors is equivalent to computation of the smallest subset $V_2' \in V_2$ such that injection of colors to this set of nodes ensures that each node in V_1 receives a unique color through *seepage.* In this study, we restrict our attention to $k = 1$ or $k = 2$ only, as cues of deteriorating health of transformer may not be observable at distances $k \geq 3$ (See Figure 10).

A variation of Identification Code when restricted to Bipartite graphs is known as *Discriminating Code* (Charon et al. 2008; Charbit et al. 2006; Charbit et al. 2008), and is defined as follows: Let $G = (V_1 \cup V_2, E)$ be an undirected bipartite graph and let $N(v)$, denote the neighborhood of v, for any $v \in V_2$, a subset $V_2' \subseteq V_2$ is called the Discriminating Code of G if $\forall v \in V_1, N(v) \cap V_2'$ is unique. The problem of determining the *minimum* cardinality Discriminating Code Set is called the Minimum Discriminating Code Set (MDCS) problem. We will refer to critical equipment health monitoring problem, with the fewest number of sensors, as the *Monitoring Critical Equipment* (MCE) problem, which may be stated formally in the following way:

MCE Problem: Find the smallest subset $V_2' \subseteq V_2$, such that injection of colors at these nodes, ensures that each node $v \in V_1$, receives a unique color through seepage.

4.2.1 Problem Formulation

In this section, we formalize the problem of computing the fewest number of sensors to be deployed to monitor all critical equipments (HV transformers) in the power grid, so that, if they show signs of potential failure, then an operator in the control room, can uniquely identify them. Once the failing equipment is identified, corrective measures can be undertaken, such as a planned shutdown.

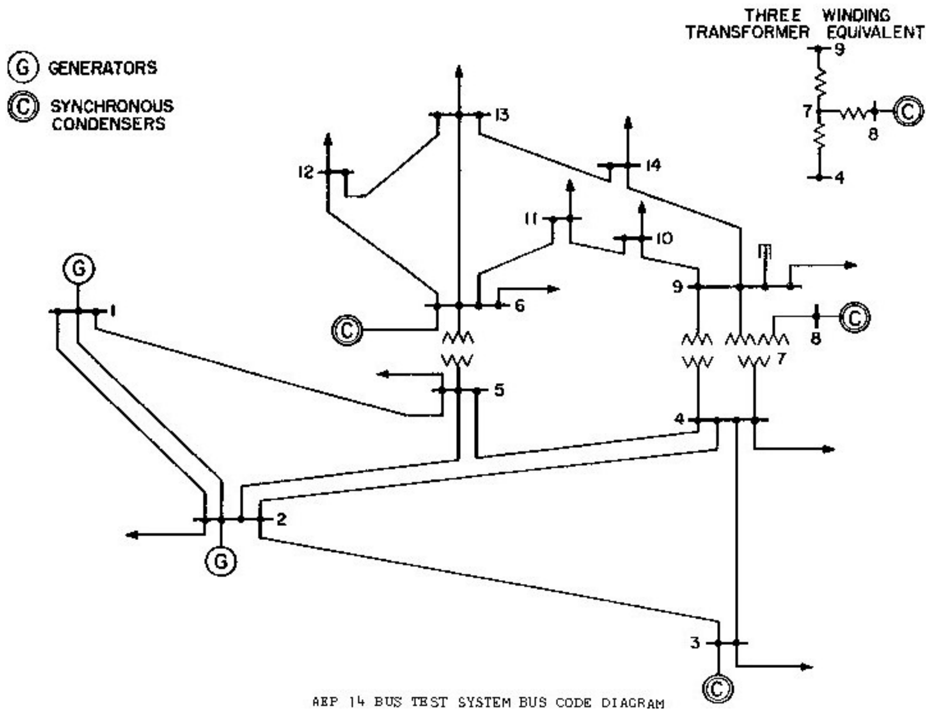

GENERATORS
SYNCHRONOUS CONDENSERS
THREE WINDING TRANSFORMER EQUIVALENT
AEP 14 BUS TEST SYSTEM BUS CODE DIAGRAM

Figure 11. IEEE 14 Bus Test System

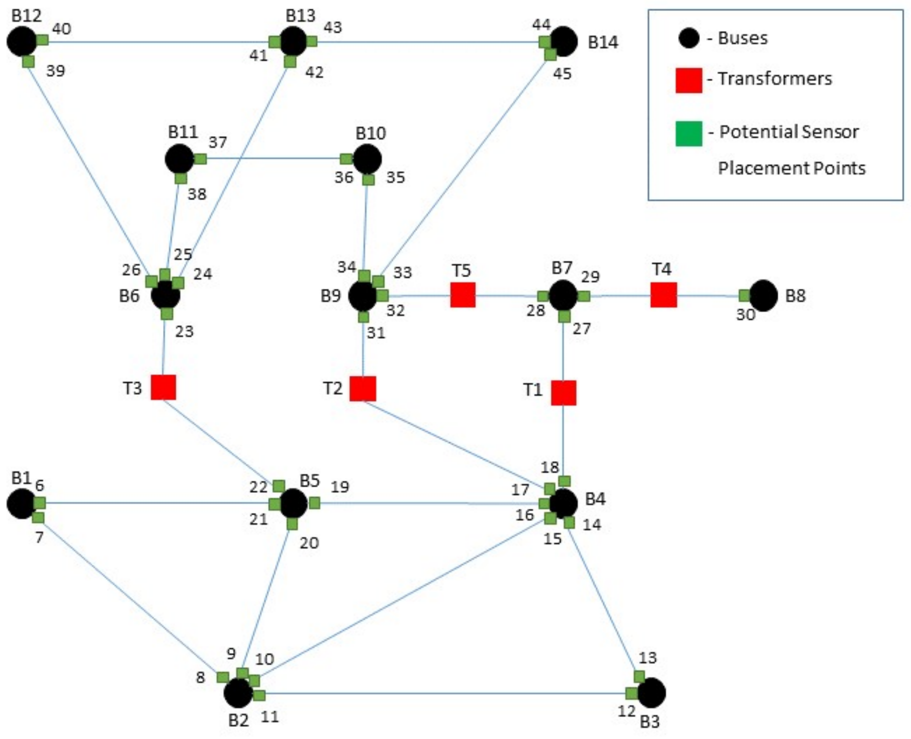


Figure 12. Potential Sensor Placement Locations in IEEE 14 Bus Test System

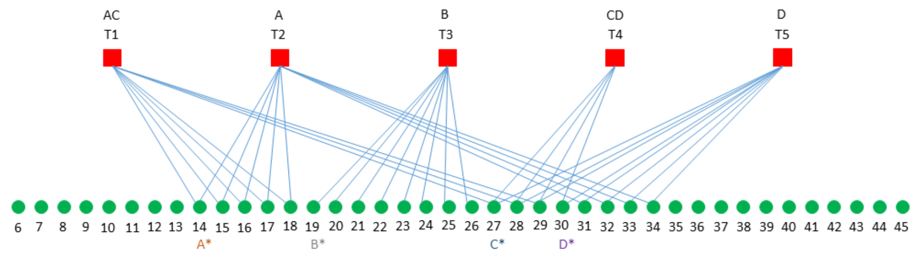


Figure 13. Bipartite Graph corresponding to IEEE 14 bus system with for k = 1

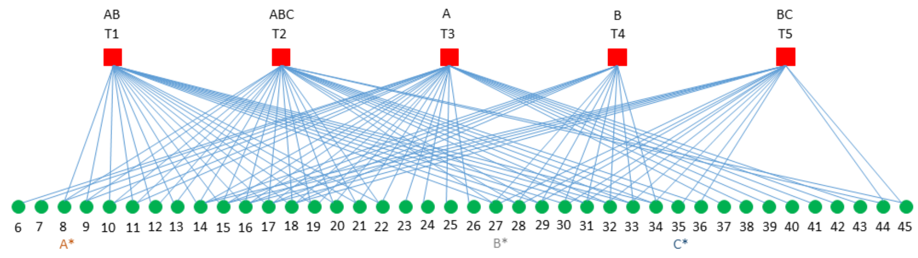


Figure 14. Bipartite Graph corresponding to IEEE 14 bus system with for k = 2

From our prior discussions, it is clear that Identifying Code relates to an underlying graph. In order to use Identifying Code to find the fewest number of sensors to be deployed to monitor critical equipments, we first have to construct a graph from the single line diagram (SLD) of the power system. Consider the IEEE 14 Bus System shown in Figure 11. We construct a graph $G = (V, E)$ from the SLD, where each node represents either a bus or a transformer, and two nodes are connected by an edge if the corresponding buses, or bus and transformer are connected. The Figure 12 shows the graph $G = (V, E)$ constructed from the IEEE 14 Bus SLD, shown in Figure 11. In Figure 12, the buses are represented by black circular nodes and the transformers by red square nodes. In power systems, the monitoring devices (such as the PMUs) can be placed on the ends of the transmission lines, next to the buses (Pal, Vullikanti, and Ravi 2016). In Figure 12, the potential locations where a monitoring device can be deployed are shown by small green squares.

The objective here is to determine the health of the red squares (transformers) before they reach a critical state. Signal of failing health of a red square reaches only up to a certain distance from the location of the red square, where distance is measured in terms of number of hops. The monitoring devices can only be placed at the green squares. If we assume that the signal of failing health of a red square can reach k hops, then all green squares within k hop distance of the red square will recognize that that particular red square (transformer) is failing. This can be captured in a bipartite graph $G = (V_1 \cup V_2, E)$, where each node $v \in V_1$ represents a red square and each node $v \in V_2$ represents a green square. There is an edge $e \in E$ connecting nodes $v_i \in V_1$ and $v_j \in V_2$ if the signal from the red square r_i, represented by node v_i in Figure 12, can reach the green square g_j, represented by node v_j in Figure 12. Such graphs corresponding to the IEEE 14 Bus System, with $k = 1$ and $k = 2$, are shown in Figure 13 and Figure 14, respectively. Since the IEEE 14 Bus System has 5 transformers (red squares in Figure 12), the vertex set V_1 in the bipartite graphs shown in Figure 13 and Figure 14 has 5 nodes. Additionally, in the IEEE 14 Bus System, there are 40 potential locations for placement of sensors (green squares), in Figure 12, the vertex set V_2, in the bipartite graphs shown in Figure 13 and Figure 14, has 40 nodes (numbered from 6-45), denoted by green circles. It may be noted that when $k = 1$, only 21 out of 40 potential locations are viable locations for placement of sensors as the other 19 locations are not within 1-hop neighborhood of the transformers. However, when $k = 2$, all 40 nodes are viable locations for placement of sensors, as all of them are within the 2-hop neighborhood of the transformers. It may be noted that some of the nodes in Figure 13 and Figure 14, are labeled with strings such as “A”, “AC”, etc. The explanation and significance of these strings are given in the experimental section.

4.2.2 Problem Solution

In this section, we provide an Integer Linear Programming (ILP) formulation for solving the MCE problem, as stated below.

Instance: $G = (V_1 \cup V_2, E)$, an undirected bipartite graph.

Problem: Find the smallest subset $V_2' \subseteq V_2$, such that injection of colors at these nodes, ensures that each node $v_i \in V_1$, receives a unique color (either atomic or composite) through seepage.

We use the notation $N(v_i)$ to denote the neighborhood of v_i, for any $v_i \in V_1 \cup V_2$. Corresponding to each $v_i \in V_2$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: *Minimize* $\sum_{v_i \in V_2} x_i$

Coloring Constraint: $\sum_{v_i \in N(v_j)} x_i \geq 1, \quad \forall v_j \in V_1$

Unique Coloring Constraint: $\sum_{v_i \in \{N(v_j) \bigoplus N(v_k)\}} x_i \geq 1, \quad \forall v_j \neq v_k, \in V_1$

$N(v_j) \bigoplus N(v_k)$ denotes the Exclusive-OR (symmetric set difference) of the node sets $N(v_j)$ and $N(v_k)$. It may be noted that the objective function ensures that the fewest number of nodes in V_2 are assigned a color. The Coloring Constraint ensures that every node in V_1 receives at least one color through seepage from the colors injected at nodes in V_2. A consequence of the Coloring Constraint is that, a node in V_1 may receive more than one color through seepage from the colors injected at nodes in V_2.

The Unique Coloring Constraint ensures that, for every pair of nodes (v_j, v_k) in V_1, at least one node in the node set $N(v_j) \bigoplus N(v_k) \subseteq V_2$ is injected with a color. This guarantees that v_j and v_k will not receive identical colors through the color seepage from the nodes in V_2.

4.2.3 Experimental Results and Discussion

Here, we present the results of of our technique on standard power system test cases, such as IEEE 14, 30, 57, 118, PEGASE 89 bus, and Polish 2383 bus systems. As discussed previously, the IEEE 14 bus system has 5 transformers and 40 potential locations for placement of sensors. The bipartite graphs for the IEEE 14 bus system for $k = 1$ and $k = 2$ are shown in Figure 13 and Figure 14. Our results obtained from the solution to the ILP show that the 5 transformers can be monitored with 4 sensors when $k = 1$, and 3 sensors when $k = 2$. As shown in Figure 13, for $k = 1$, if 4 sensors are deployed at nodes 14, 19, 27, and 30 (equivalently 4 colors A, B, C, and D are injected at these nodes, shown in Figure 13 by A*, B*, C*, and D*), the 5 transformers T1 through T5 will receive *unique* colors AC, A, B, CD, and D, respectively. Similarly, for $k = 2$, if 3 sensors are deployed at nodes 8, 27, and 35 (equivalently 3 colors A, B, and C are injected at these nodes, shown in Figure 14 by A*, B*, and C*), the 5 transformers T1 through T5 will receive *unique* colors AB, ABC, A, B, and BC respectively.

The significance of each transformer receiving a unique color (or a unique signature), is the following. In the example shown in Figure 14, if colors A, B and C are injected at nodes 8, 27 and 35 (i.e., PMUs A, B, and C are placed at these locations, among the 40 (6-45) potential locations), the transformers T1-T5 will receive colors AB, ABC, A, B, and BC, respectively. Suppose that the control room has three indicator lamps, 1, 2, and 3, corresponding to PMUs A, B, and C, respectively. As long as the width of the SNR ratio is within the normal range, the lamps are green. As soon as the width of the SNR ratio exceeds the normal range, the corresponding lamps turn red. An operator, at the control room, can interpret the status of the five transformers, in the following way: (i) The transformer T1 is failing if only lamps 1 and 2 turn red, (ii) T2 is failing if lamps 1, 2 and 3 turn red, (iii) T3 is failing if lamp 1 turns red, and so on.

Bus System	No. of Transformers	No. of Sensors	
		$k = 1$	$k = 2$
IEEE 14	5	4	3
IEEE 30	7	6	4
IEEE 57	14	13	10
PEGASE 89	10	10	6
IEEE 118	9	9	5
Polish 2383	155	155	106

Table 19. No. of sensors needed in IEEE, PEGASE, and Polish systems for $k = 1, 2$.

Our results for power system test cases are tabulated in Table 19. The results show that the number of sensors needed to monitor all the transformers are fewer than the number of transformers. On an average there were 6.90% and 37.90% savings in the number of sensors using our technique for $k = 1$ and $k = 2$, respectively. From Figure 10, it can be seen that the difference in the width of the SNR band in dB at substations S_1 and S_2 (1 and 2 hop distance away respectively, from the transformer) is minimal. Accordingly, we can use $k = 2$ results, which implies that significant savings (37.90%) can be realized using our technique. The ILPs for the test cases were computed using GUROBI for python. An Intel Core i5-6300HQ CPU with 2.30 GHz and 32 GB RAM was used for our experiments. The computation time varied from 0.17 seconds, for the smallest test case ($|V_1| = 5$, $|V_2| = 40$, $|E| = 36$, $k = 1$), to 25.18 seconds ($|V_1| = 155$, $|V_2| = 5{,}772$, $|E| = 3{,}655$, $k = 2$) for the largest. As the computation times for these test cases were only a few seconds, we expect that for larger systems involving thousands of buses and hundreds of transformers, the problem can still be solved within a short period of time.

4.3 Robust Monitoring of Electric Grid Transformers in Adversarial Environments

Conceptually, Moving Target Defense (MTD), popular in cyber-security, seeks to continuously move between a set of system configurations available to a defender, to take away the attacker's advantage of reconnaissance (Jajodia et al. 2011). The key idea is that the attacker may not encounter the expected system configuration at the time of the attack, thereby being rendered ineffective. Formally, an MTD system can be described using the three-tuple $\langle C, T, M \rangle$, where C represents the set of system configurations a defender can move between, T represents a timing function that describes when the defender moves and M represents the movement strategy (Sengupta, Chowdhary, Sabur, et al. 2020). The goal of this work is two-fold– (i) to construct a desirable set C (for which we define the K-δMDCS problem in subsection 4.3.1) and (ii) an optimal movement strategy M (by modeling the interaction as a game in subsection 4.3.3).

Note that, when a single attack can cripple all the defense configurations $\in C$, MTD cannot aid in improving the robustness. In (Sengupta, Chakraborti, and Kambhampati 2019), the authors introduce the notion of *differential immunity* that aims at measuring the amount of diversity between configurations $\in C$. In this work, we consider a C that is differentially immune (denoted as δ), i.e. each attack, allowed by the threat model defined later, can only cripple one defense configuration. This ensures maximum diversity of C and implies the highest robustness gains for the formulated MTD.

4.3.1 K Differentially Immune MDCS (K-δMDCS)

To design the configuration set C for an MTD system, we first need to find multiple MDCS sets of a bipartite graph. For this purpose, we desire K differentially immune MDCS (K-δMDCS) where no two MDCS solutions share a common sensor placement point. Formally,

Definition 4.3.1. (K-δMDCS) Given a Bipartite Graph, $G = (T \cup S, E)$, K vertex sets $S_i \subseteq S, i \in \{1, \ldots, K\}$ are defined to be K-δMDCS of G, if the following conditions hold– (1) all the sets S_i are MDCSs of graph G and (2) for all possible pairs of sets (S_i, S_j), $S_i \cap S_j = \emptyset$.

First, we want to activate the minimum number of sensors placed in the network, for unique monitoring of HVTs, at any point in time. Hence, we use K sets, all of which are MDCS, i.e. have the smallest cardinality. Second, the use of differentially immune MDCS attempts to optimize for robustness in adversarial settings. If an attacker were to attack a particular sensor $s \in S$, it can hope to, at best, cripple a singular MDCS $S_i \in C$, from uniquely identifying HVT failure. If the defender selects an MDCS $S_j \in C (j \neq i)$, then the attacker will not succeed in affecting the functionality of the power grid sensors. We will now show that the decision problem corresponding to K-δMDCS is NP-complete.

Lemma 2. *K-δMDCS is NP-Complete, given K is an integer and $K > 0$.*

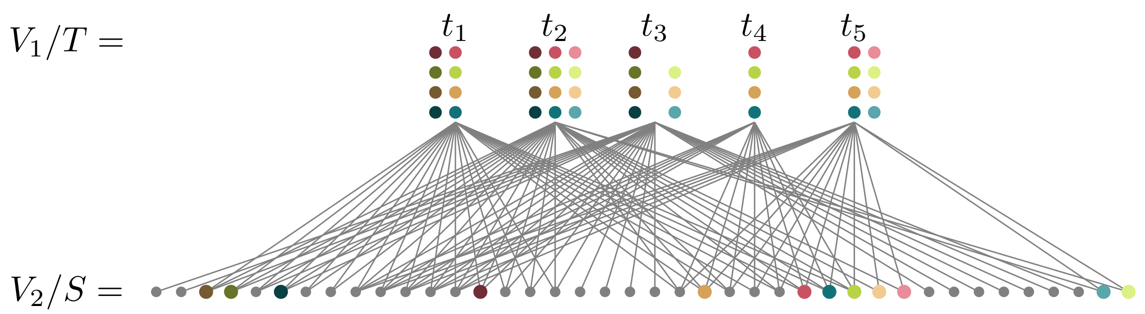


Figure 15. The IEEE 14-bus power grid graph has $4-\delta$MCDS solutions.

Proof. We note that the original MDCS problem, which is known to be NP-Complete (Charbit et al. 2006), is a special case (when $K = 1$). $\square$

Corollary 2.1. *K-δ Graph Problems such as K-δMinimum Identifying Code Set (MICS), K-δMinimum Set Cover (MSC), K-δMinimum Vertex Cover (MVC) is at least NP-Hard when K is an integer and $K > 0$.*[1]

Let us denote the size of an MDCS for a bipartite graph G as m. In K-δMDCS, the goal of the defender is to find K MDCSs each of size m. Then, the defender needs to place $K * m$ sensors in the power grid and, at any point in time, activate an MDCS set (of size m) to uniquely identify failures in T. While a large number of defender strategies (i.e. larger values of K) helps to increase their options for sensor activation, in turn reducing the success rate for the attacker, it also incurs the cost of placing $K * m$ sensors. Thus, the ideal choice of K should trade-off robustness *vs.* sensor costs (note that, when $K = 1$, robustness using MTD is impossible to achieve).

In cases where the defender has sufficient resources, one might ask *what is the maximum size of K?* Depending on the structure of the underlying graph, this question may have a trivial answer. For example, if the bipartite graph has a $t \in T$

[1]Note that in the context of these problems, the distinction between the node sets T and S in MDCS are unnecessary and one can view the graphs as $G = (V, E)$.

and $N(t) = \{s\}, s \in S$, any MDCS of G needs to place a sensor on s to uniquely detect a fault in t. Hence, there can exist no two MDCSs that do not share a common node since s has to be a part of both. In such cases, the max value of K, denoted as $K_{\max}$, is 1. Beyond such cases, similar to the problem of finding the maximum value of K in the K-clique problem, finding $K_{\max}$ demands a search procedure over the search space of K that we now describe.

4.3.2 Finding max K for K-δMDCS

We first propose a Quadratically Constrained Integer Linear Program (QCILP) that given a value of K, finds K Discriminating Code Sets (DCSs). We then showcase the algorithm for searching over possible values of $k \in \{1, \ldots, |S|\}$ to find the largest K. To define the QCILP for $G = (T \cup S, E)$, we first consider $|S| * k$ binary variables where, $x_{sk} = 1$ if a sensor is placed in node $s \in S$ for the kth DCS, and 0 otherwise. We also use a variable l that denotes the size of the DCSs found. We can now describe our QCILP, presented below.

$$\begin{aligned}
&\min_{l,x} && l && && (4.2)\\
&s.t. \quad l &=& \sum_s x_{sk} \quad \forall k && \text{All } k \text{ DCS has the same size } l.\\
&\sum_{s\in S}(x_{sk} - x_{sk'})^2 &=& 2l \quad \forall (k,k') && \text{No two DCSs should have a common sensor.}\\
&\sum_{s\in N(t)} x_{sk} &\geq& 1 \quad \forall t, \forall k && \text{All } t \in T \text{ has a sensor monitoring them for all the } k \text{ solutions.}\\
&\sum_{s\in N(t)\bigoplus N(t')} x_{sk} &\geq& 1 \quad \forall (t,t'), \forall k && t \text{ and } t' \text{ trigger unique sensors for the } k\text{-th DCS.}\\
&x_{sk} &\in& 0, 1 \forall s, \forall k
\end{aligned}$$

The last two constraints ensure that each of the K solutions are Discriminating Code Sets where, (i) all $t \in T$ trigger at least one sensor $s \in S$ and, (ii) for all pairs of t and t' (both $\in T$), there exists at least one sensor in the symmetric difference set of t and t' that is a part of the DCS, which in turn uniquely distinguishes between t and t'. The first two constraints ensure that all k DCSs are of equal size and no two DCSs shares a common sensor. We can now ask the question as to whether the DCSs found by Equation 4.2 is indeed the Minimum DCSs (MDCSs) for the graph G. In this regard, we now show the following.

Theorem 3. *For all values $K \leq K_{\max}$, the optimization problem in Equation 4.2 returns K-δMDCS.*

Proof. We consider proof by contradiction. Given the value of $K (\leq K_{\max})$, let us assume that the solution returned by Equation 4.2 is not the K-δMDCS for the graph G. If this is the case, at least one of the two properties in the definition of K-δMDCS is violated. Thus, either (i) the returned solution consists of a DCS that is not the Minimum DCS, or (ii) there exists a sub-set (of size greater than one) among the set of DCSs that share a common node.

Owing to the third and fourth constraints, all the solutions constitute a DCS. Now, if (i) is violated, all the DCSs returned by the QCILP, of length l, are not the MDCS for G. Thus, the MDCS must have a DCS of size $l' \leq l$. Given that the minimization objective finds the smallest DCS and $K \leq K_{\max}$, this cannot be possible. Hence, (i) does not hold.

For (ii), let us say that there exists a subset of the DCSs returned that share a common node. If this was the case, then at least one solution pair has to share a common node. If this node is denoted as s^* and the two solutions are termed as k and k', then for the second constraint, given $x_{s^*k} = x_{s^*k'} = 1$, the term for s^* is zero.

Algorithm 1 Finding $K_{\max} - \delta$MDCS.

```
1: In: $G = (T \cup S, E)$
2: Out: $K_{\max} - \delta$MDCS
3: solutions $\leftarrow \emptyset$
4: $K \leftarrow 1$
5: while $K \leq |S|$ do
6:     solutions$_K \leftarrow$ Solve Equation 4.2 with $K$
7:     if solutions$_K == \emptyset$ then
8:         break                                  Infeasible for $K > K_{\max}$
9:     end if
10:    if solutions $! = \emptyset$ and |solutions$(l)$| < |solutions$_K(l)$| then
11:        break                                  DCS returned is not MDCS for $K > K_{\max}$
12:    end if
13:    solutions $\leftarrow$ solutions$_K$
14:    $K \leftarrow K + 1$
15: end while
16: return solutions
```

Even if the other $l - 1$ nodes in the solutions k and k' are unique, the terms will add up to $2 * (l - 1)$ thereby violating the second constraint. This is not possible and as a consequence, (ii) does not hold. □

Given this, we can now consider cases where $K > K_{\max}$. When $K > K_{\max}$, the optimization problem in Equation 4.2 is either infeasible or returns K DCSs that are not MDCS for graph G. This condition holds by the definition of $K_{\max}$ (proof by contradiction ensues if neither of the two cases holds). With these conditions in mind, we can design an iterative approach, shown in Algorithm 1, to find the $K_{\max} - \delta$MDCS of a given graph.

Figure 15 showcases the $4 - \delta$MDCS solutions returned by Algorithm 1 for the 14-bus power grid network. The different colors indicate the different MDCSs found for G and the shades of the same color indicate an MDCS set. As shown, each of the four MDCS has a size of $l = 3$ and uniquely identifies all the transformers T. The lack of overlapping colors in the bottom set of nodes indicates that no two MDCS share a common $s \in S$.

While the procedure in Algorithm 1 finds the $K_{\max} - \delta$MDCS, it can be time-consuming for the largest networks (although it works well on large power-grids as shown in the experimental section). Thus, one can consider a greedy approach in which, one solves the MDCS problem using (Basu, Padhee, et al. 2018). We then solve this ILP with the additional constraints that $x_s = 0$ for all the sensors found in the current solution and keep doing so until (i) the ILP becomes infeasible or (ii) results in DCS that does not have minimum cardinality. In the experimental section, we will see that although this approach is faster, it can output K-δMDCS where $K < K_{\max}$. The sub-optimality is a result of the ordering "enforced" by the current optimal MDCSs which in turn, proves to be infeasible constraints for the latter iterations of the problem.

4.3.3 Game Theoretic Formulation

The defender's goal is to maintain the unique identifying capability of HVTs at all times. Conversely, the attacker tries to prevent this capability, thereby making it harder for the defender to effectively monitor the HVTs. Here, we seek to find the optimal movement function M for the sensor activation MTD to aid the defender to realize its objective. To do so, we consider a strong threat-model where the attacker $\mathcal{A}$ with recon, is aware of the defender $\mathcal{D}$'s (probabilistic) sensor activation strategy, thereby making the Stackelberg Equilibrium an appropriate solution concept for our setting. We use a polynomial-time approach to find the Strong Stackelberg Equilibrium of the game (Conitzer and Sandholm 2006). We now briefly describe the various parameters of the formulated game (see Figure 16).

Defense Actions: The defender has $K_{\max}$ pure strategies and the configuration set $C = K_{\max} - \delta$MDCS. If one uses the greedy algorithm instead of the optimal approach

$$
\begin{array}{c|cccc}
\downarrow A^{\mathcal{D}} \mid A^{\mathcal{A}} \rightarrow & ● & ● & \cdots & ● \\
●●● & \begin{array}{c} 0, \\ U^{\mathcal{D}}(\binom{t_1}{t_4}, \binom{t_2}{t_5}, t_3) - C^{\mathcal{A}}(●) \end{array} & & \cdots & \begin{array}{c} +\sum_{t \in T} U^{\mathcal{D}}(t), \\ -C^{\mathcal{A}}(●) \end{array} \\
\begin{array}{c} ●●● \\ ●●● \end{array} & \cdots & & \ddots & \cdots \\
●●● & \begin{array}{c} +\sum_{t \in T} U^{\mathcal{D}}(t), \\ -C^{\mathcal{A}}(●) \end{array} & & \cdots & \begin{array}{c} U^{\mathcal{D}}(t_3), \\ U^{\mathcal{D}}(\binom{t_1}{t_2}, \binom{t_4}{t_5}) - C^{\mathcal{A}}(●) \end{array}
\end{array}
$$

Figure 16. Game-matrix for the dynamic sensor activation problem.

(both described in the previous section), the number of pure strategies obtained may be less than $K_{\max}$.

Attack Actions: We assume that an attacker can spend reconnaissance effort in figuring out the sensor placement point. Thus, its action set includes attacking a sensor that may be considered for activation (instead of all nodes in $|S|$). While one can consider attackers with the capability to attack multiple sensor activation points, it is often too expensive a cost model as it demands resource procurement and distribution over a wide geographic area.

Player Utilities: The game has two different kinds of utilities that are used to calculate the rewards. First, the defender receives the utility associated with uniquely identifying a transformer $t \in T$ in the case of anomalous spikes indicative of failure (to occur). We assume that a transformer supplying power to an important building (eg. the White House or the Pentagon) is considered to be more important than one supplying power to a residential area. Second, the attacker's cost for attacking a particular sensor needs to be considered. While some sensors may be placed in high-security areas, others may be easier to access. We conduct randomized trials

		C		Movement Function M			
Graph	$\lvert S\rvert + \lvert T\rvert$	$\lvert A^{\mathcal{D}}\rvert$ $(K/K_{\max})$	$\lvert A^{\mathcal{A}}\rvert$ $(K/K_{\max})$	URS (K)	URS $(K_{\max})$	SSE (K)	SSE $(K_{\max})$
14 Bus	45	4/4	12/12	18.5±4.7	18.65±4.7	20.62±4.6	**20.72**±4.6
30 Bus	89	4/4	16/16	26.45±5.7	27.25±5.6	29.44±6	**29.9**±5.8
39 Bus	96	7/9	28/36	18.7±5	19.24±5.2	**19.8**±5.3	19.73±5.3
57 Bus	170	6/6	60/60	70.76±10.8	70.88±11.1	**73.5**±10.6	73.07±10.7
89 Bus	422	16/21	96/126	50.67±8.9	51±9	**52.2**±9.2	**52.2**±9.2
118 Bus	367	2/2	10/10	31.35 ±6	31.6 ± 6	32.45±6.4	**32.61**±6.1
2383 Bus	5927	2/3	212/318	832.7±38.7	836.16±36.7	835.34±39	**842.34**±39.4

Table 20. Game parameters and defender's reward for playing the different Cs and Ms for the various power-grid networks.

with both these values $\in [0, 10]$, with 10 indicating the HVT/sensor most important to protect/difficult to attack.

In the bottom right corner of Figure 16, the defender, owing to the attacker attacking a sensor, is only able to uniquely identify t_3 and thus, only gets reward proportional to it. Conversely, the attacker, due to attacking a sensor, can make failures of t_1 and t_2 (and t_4 and t_5) indistinguishable and receives the corresponding utilities, minus the cost of attacking the sensor denoted by the light blue node ($\in S$, Figure 15). Similarly, if the attacker selects the attack represented by the first attack column (sensor denoted by the dark brown node), the defender cannot identify any HVT and thus, gets a utility of zero.

4.3.4 Experimental Simulation

In this section, we conduct simulation studies on seven IEEE test graphs popular in the power domain (Zimmerman, Murillo-Sánchez, and Thomas 2010). Characteristics of these graphs such as the total number of nodes (i.e. $|S| + |T|$) are shown in Table 20. The table further lists the K values for the K-δMDCS found by the greedy and the optimal Algorithm 1, and is denoted by K and K_{max} respectively. The number of attacker strategies is listed in the fourth column. This value can be obtained by multiplying the corresponding K value with the size of an MDCS for graph G, since none of the K-δMDCS share a common node. We now discuss two results– (i) the effectiveness of the game-theoretic equilibrium compared to the Uniform Random Strategy baseline (which chooses to activate a particular MDCS with equal probability) and (ii) the time is taken by the greedy and the optimal algorithm and their respective solution quality. [2]

4.3.4.1 Effectiveness of Game-Theoretic Equilibrium

In Table 20, we show that in all test cases, the optimal movement strategy at the Strong Stackelberg Equilibrium (SSE) gives the defender a higher reward than choosing Uniform Random Strategy (URS). When using URS or SSE, in most cases we see higher gains when the construction of the MTD configuration set C is optimal ($URS(K_{\max})$ obtained from Algorithm 1) as opposed to using a greedy algorithm ($URS(K)$). We expected this as the higher number of differentially immune options (as $K_{\max} > K$) chosen with equal probability reduces the probability of picking the weakest strategy. When the value of $K_{\max} = K$, such as for 14, 30, 57 and 118 buses, we see that the difference between the two versions of URS (or two versions of SSE) are negligible. A reason for the non-zero difference between the rewards values arises because of the MDCS sets chosen, although the total number of sets chosen are the same. We also see that the difference in defender rewards can be large even when the difference between K and $K_{\max}$ is small in the case of larger networks (eg. 2383

[2]The code for the experiments can be found at https://github.com/kaustav-basu/Robust-MICS

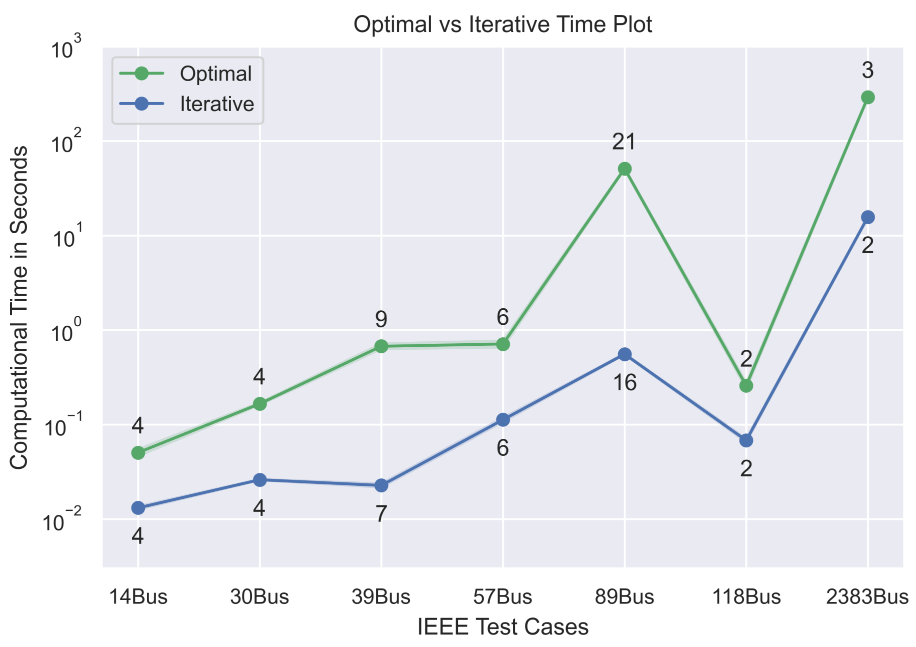


Figure 17. Time taken by the optimal vs. the greedy approach for finding (the K values are shown above the plot points).

bus). Thus, without finding the $K_{\max}$ and the SSE for the optimal C, it is hard to establish the loss in rewards. Given that these strategies are pre-computed, the power grid utility operator should not consider the greedy strategy unless the time required becomes prohibitive.

4.3.4.2 Computational Time for finding C

In Figure 17, we compare the time taken for finding the configuration set C using the optimal *vs.* the greedy approach. We choose the logarithmic scale for the y-axis because the computational time of the optimal and greedy approaches for the 14, 30, 39, 57, and 118 buses was less than a second, and thus difficult to distinguish between on a linear scale. The largest disparity occurs when the size of the optimal set $K_{\max}$ is greater than the K-sized set found by the greedy approach (39/89/2383 Bus). In other cases, while the optimal approach is slower, it provides the guarantee that no solution with a greater K exists, which is absent in the greedy case. A case where the logarithmic scale, from a visualization perspective, does not do justice is the 2383-Bus. The time taken by the greedy approach is $15s$ compared to $291s$ taken by the optimal approach. While the K value differs by a factor of one, the resultant gain in defender's game value, as shown in Table 20, is relatively large. Thus, the added time in generating the optimal configuration set needs to be criticized based on the gain obtained in the underlying game.

We also consider the pragmatic scenario when the K value is fixed by the defender up-front owing to budget restrictions of sensors that can be placed in the power network. In this case, the greedy approach has to iteratively find one solution at a time, adding them to the constraint set of future iterations until the desired k is reached. On the other hand, the iterative procedure in Algorithm 1 can be altogether ignored and one can simply return the solution found by the optimization problem in Equation 4.2.

4.4 Future Research Directions

This chapter dealt with the objective of monitoring critical entities (High Voltage Transformers) in a power grid system. We showed how a variant of the Identifying Code, called the Discriminating Code, can be effectively utilized for uniquely monitoring these critical entities. Moreover, we considered the adversarial scenario, where the attacker can attack at most one entity. That being said, however, there are multiple avenues which could be examined further in this domain. To highlight a few:

- *Generalized Adversarial Scenario*: In this chapter, we analyzed the scenario where the attacker can attack only one target sensor. However, with increasing computational power, it has become feasible for attackers to launch simultaneous attacks. Therefore, a more realistic and generalized approach is necessary to accurately model such a scenario.
- *Fault Tolerance*: In this chapter, we examined robustness in terms of immunity from adversarial attacks. However, we have not considered the scenario where, (i) the deployed sensors in the power grid fail (either due to adversarial attack or mechanical failure), or (ii) the components in the power grid (buses, transmission lines) fail. Such a setting dictates the deployment of additional sensors in the power grid, thereby increasing redundancy in the system. As a result, the development of algorithms for such fault tolerance is necessary.
- *Budget Constraints*: For both the problems addressed in this chapter, we have neglected a real world constraint, the budget of the power grid utility operator. For certain cases, it may not be feasible to procure and deploy all the sensors necessary for unique monitoring of the entities. Therefore, the problem of determining the judicious placement of k sensors ($k < m$, where m is the cardinality of the MDCS solution), such that the maximum number of entities are uniquely monitored, should be explored.

Chapter 5

ONLINE SOCIAL NETWORKS

In this chapter, we focus on the identification of the sources of anomalies propagating on online social networks. The anomaly studied here is misinformation propagation, however, it should be noted that any anomaly which can propagate in a social network can be uniquely monitored using our approach. Researchers have studied the dissemination of misinformation on social networks and attempted to identify its source(s) for the better part of the last decade. Numerous studies assumed that misinformation disseminated in a manner similar to that of infectious diseases and, as a result, utilized models such as SI, SIR, SEIS, etc. in order to place sensors and identify the source of the dissemination (Shah and Zaman 2011; Zhu, Chen, and Ying 2016; Y. Zhou et al. 2019). However, these approaches suffer from certain drawbacks. Firstly, these works assume the existence of an underlying spreading model. In the real world, each misinformation may disseminate differently and the selection of an appropriate model may become difficult, thereby hindering the task of successfully identifying the source of the dissemination (Z. Wang et al. 2017; Dong et al. 2019). Secondly, even if we assume the existence of an underlying spreading model, these works further assume that at the end of the spreading process, all the nodes in the network have the capability to declare if they have been infected with misinformation or not (Spinelli, Celis, and Thiran 2017). This is only possible if the nodes themselves are fact checkers, which is not the case on social networks. Finally, even if one may correctly assume a dissemination model, it is hard to estimate the values of the parameters of the model (Z. Wang et al. 2017; Dong et al. 2019). Thus, there is clearly a need for the development of misinformation source detection algorithms which do not rely heavily on an underlying dissemination model.

In this chapter, we attempt to fill in this gap by presenting an approach based solely on the topological structure of a social network. We assume that we have complete knowledge of the social network under study and our graph theoretical framework utilizes this in order to place the "sensors" [3] on a *minimum subset of the nodes in the graph*, to accurately identify the users engaging in misinformation dissemination. We show that by following our approach, there is no need to actively monitor each and every user, but only a *subset* of the total users set, which could be as low as $O(\log n)$. *In other words, unlike the epidemiological models, only a small subset of the users have to be fact-checked, as opposed to everyone in the network.* Moreover, we show that by monitoring the users (nodes) in this subset, our approach can still *uniquely identify any user in the social network, who engages in misinformation dissemination.* Our resource optimized approach exploits a common characteristic of social networking platforms: social network posts/comments of an individual are "visible" to his/her immediate friends (assuming that the post does not have custom visibility settings). We show that our approach, based on the mathematical concept of *Identifying Codes*, reduces the resource requirements (fewer number of users to be monitored and as a direct consequence, lesser computational time) significantly for the platforms, with the help of extensive experimentation on anonymous real world Facebook datasets.

[3] We would like to point out here that, in this chapter, we focus primarily on the placement of these detection sensors and not on their development.

5.1 Related Work

Here, we highlight the motivating studies behind our work in three broad areas - development of the detection sensors, epidemiological model based misinformation source identification and the unique identification capabilities of Identifying Codes.

5.1.1 Detection Sensors

Preliminary research on misinformation detection was primarily unimodal (either textual or visual). Textual approaches, by (Potthast et al. 2017), considered features such as headlines, lexical, syntactic, semantic, writing style etc. of social media posts to determine if the content was information or misinformation. (Gupta et al. 2013) studied visual approaches and tried to identify certain features which can be utilized for the classification of images as information or misinformation. Recently, (Jin et al. 2017; Y. Wang et al. 2018; Khattar et al. 2019; Qi et al. 2019) utilized deep neural networks to classify multi-modal social media posts (textual + visual). Further studies focused on the social context aspect to determine the authenticity of social media posts. (Shu, Wang, and Liu 2018) analyzed user profiles, (Yang et al. 2019) analyzed user opinions in an unsupervised manner and (Jin et al. 2016) studied the user opinions towards social media posts to determine the veracity of the post. The problem of fake news mitigation was mapped to the reinforcement learning framework, with the goal of optimizing the actions for maximal total reward under budget constraints by (Farajtabar et al. 2017). (Shi and Weninger 2016), viewed link-prediction task in a knowledge graph to accurately determine the veracity of a fact. (Shu et al. 2017) presented a survey of detecting fake news on social media. Real-world datasets measuring users' trust level on fake news was constructed by (Shu, Wang, and Liu 2018). (Tacchini et al. 2017) showed that Facebook posts can be classified with high accuracy as hoaxes or non-hoaxes on the basis of the users who "liked" them.

5.1.2 Epidemiological Models for Source Detection

The seminal work of (Shah and Zaman 2011) paved the foundation for the rumor source detection problem. In their work, they assumed that rumors spread according to the SI model on social networks and that they had complete information about all the states of the nodes (including network parameters) in the network. (Zhu and Ying 2014) also assumed that they had complete information about all the node states but assumed that rumors spread according to the SIR model for detecting single rumor sources. In their follow up work, (Zhu, Chen, and Ying 2016) swapped their assumption of the complete network snapshot with that of the partial network snapshot and present two algorithms to detect *multiple* diffusion sources. (Spinelli, Celis, and Thiran 2017) presented static and dynamic sensor placement approaches for rumor source detection, wherein the rumor disseminates following the SI model. (Paluch et al. 2020) go one step further and assume that each node has the capability of reporting which neighboring node sent the virus (misinformation). (Tang 2020) argued that it is difficult to know the topology of the network in advance and utilized network topology inference and epidemiological models for the detection of the sources of misinformation. (Racz and Richey 2020) studied the problem of robust rumor source identification problem in the face of adversaries, who can perturb the original misinformation in order to shield the source. Even though there exists numerous works on epidemiological model based rumor source(s) identification, authors in (Dong et al. 2019; Z. Wang et al. 2017) criticized the epidemiological model assumptions and stated that knowing the underlying model beforehand is infeasible. (Z. Wang et al. 2017) presented an approach to propagate (without knowing the underlying model) the infection label throughout the network and use peaks to identify the source nodes. (Dong et al. 2019) builds on (Z. Wang et al. 2017) and utilizes Graph Convolutional Neural Networks to identify sources based on non-integral node infection labels.

5.1.3 Identifying Codes

Sensor placement optimization for the unique identification of the nodes in a graph was first introduced as Identifying Codes by (Karpovsky, Chakrabarty, and Levitin 1998) and provided results for a class of graphs. (Irène Charon, Hudry, and Lobstein 2003) proved the NP Completeness for the minimum Identifying Code problem based on a reduction from the 3-SAT problem. (Ray et al. 2003) introduced the concept of robust Identifying Codes to deal with faults in sensor networks. The Identifying Code problem was approximated to an approximation factor of $O(\log n)$ in (Gravier, Klasing, and Moncel 2008; Suomela 2007) by utilizing the notions of entropy and disjoint unions. It was shown in (Moncel 2006) that for a particular class of graphs, the cardinality of the Identifying Code solution could be as low as $\log_2\ n + 1$. Integer Linear Programs to compute the Minimum Identifying Code Set (MICS) for a given graph was presented in (Basu, Padhee, et al. 2018; Basu, Zhou, et al. 2018; Basu and Sen 2019a, 2019b, 2021b; Basu et al. 2019) for monitoring networks obtained from numerous domains such as terrorism, drug and critical infrastructures. A lower bound on the MICS for a k-fault tolerant system was presented in (Arunabha Sen et al. 2018). It was shown in (Basu 2019) how the same ILPs can be utilized for the computation of the minimum number of users in order to uniquely identify users engaging in misinformation dissemination.

Here, we show that the *ILP based approaches fail in case of social networks and accordingly, present approximation and scalable minimal algorithms.* Additionally, our work is different from the motivating works of (Z. Wang et al. 2017; Dong et al. 2019) in the sense that both works take as input a set of nodes who have already been infected and make a reasonable assumption that the nodes surrounded by infected nodes are more likely to be the source nodes. In this chapter, however, we do not need to keep track of individual labels of each node in the network and can simply identify the sources of misinformation *by triangulating the classification outputs of the detection sensors placed in the network.* Finally and more importantly, our approach *does not take into account any prior information regarding the nodes which have already been infected.*

5.2 Problem Formulation

In this section, we formalize our problem of finding the minimum number of users, on whom detection sensors must be placed to ensure unique identification of all the users in the network, in case the user represented by v, engages in misinformation dissemination. In this effort, we assume that only *one* node v becomes active at a time step and each and every node $v \in V$ can be monitored.

Graph Construction Rules: A graph representing a social network can be constructed as follows: we can denote each registered user on the social networking platform by a node. If a user u is friends with another user v and posts shared by u

shows up on v's timeline and vice versa, then there is an edge between the two. If, for some reason, u and v are friends but u's posts do not show up on v's timeline due to u's custom visibility settings, then, in our work, we do not consider an edge to exist between them.

As evident, our approach does not assume the existence of an underlying epidemiological model for the dissemination of misinformation. We utilize the connections between users on the social network in order to determine the source of misinformation dissemination. Our approach relies on the social network property that posts made by a user's connections will be visible (or available) on the timeline of the user in question. Moreover, we do not assume that the nodes in our network have the capability to reveal their state (either infected with misinformation or not) after misinformation dissemination, as is evidenced by prior approaches. As mentioned previously, this assumption limits the applicability of prior approaches. In our work, we place detection sensors, which *do have the capability to distinguish between real and fake content*, on a small subset of the users in the network. One critical aspect of misinformation detection on social networks is to monitor *all the unique posts* generated by users on the platform. Thus far we have discussed the placement of a minimum number of sensors in order to uniquely monitor all the nodes in the graph. We have to guarantee that by monitoring the content of this subset of the nodes ensures that we monitor all the unique posts generated on the social networking platform. It is trivial to note that, if we place detection sensors on *all* the nodes of the graph, then all the posts generated on the platform will be definitely monitored. We now argue that even with the monitoring of a small subset of users (provided there are no custom post settings which prevent the post from propagating to all the friends), we still retain the capability of monitoring each and every unique post generated on the platform.

Theorem 4. *Given a graph $G = (V, E)$ corresponding to a social network, in which V' is an MICS, by placing detection sensors on every node $v' \in V'$, our approach monitors all unique posts generated by users on the platform for misinformative content.*

Proof. Assume that the detection sensors have been placed on a subset V' of a graph $G = (V, E)$, where V' is an MICS of the graph. It is trivial to note that these sensors can monitor the content generated by the users they have been placed on. By the definition of ICS, at least one detection sensor has been placed in the neighborhood of each and every $v \in V - V'$. If this wasn't the case then we would have nodes in the graph which would not have been monitored. Therefore, if any user u, were to post any content on their timeline, then the post would appear on the timelines of a set of users $\mathcal{V}$ on which the detection sensors have been placed, where $\mathcal{V} \subseteq N^+[u]$ and $\mathcal{V} \subset V'$. Thus we can guarantee that all the *unique* posts on the platform are monitored.

□

Hitting Set (HS) Formulation: We now provide a novel transformation where, we view the MICS problem as a Minimum Hitting Set (MHS) problem. This transformation allows us to utilize the well known greedy algorithm of the MHS problem in order to provide the approximation bound for the MICS problem. Interestingly, previous research efforts which determined the approximation bounds for the MICS problem explored convoluted routes, such as determining entropy, disjoint sets, etc. (Xiao, Hadjicostis, and Thulasiraman 2006; Gravier, Klasing, and Moncel 2008). It may be noted that the greedy heuristic for the HS problem provides a $O(log\ m)$ factor performance bound, where m is the number of elements in the collection set (Vazirani 2013). In the following, we define the minimum HS problem:

Definition 5.2.1. Given a universal set $\mathcal{U} = \{u_1, ..., u_n\}$, and a collection set $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_m\}$, where $\mathcal{S}_i \in \mathcal{U}$, find the smallest subset $U' \subseteq \mathcal{U}$, which hits every set $\mathcal{S}_i \in \mathcal{S}$.

Definition 5.2.2. Closed Neighborhood of $v_i = CN(v_i) = N^+(v_i)$, where $N^+(v_i) = N(v_i) \cup \{v_i\}$, where $N(v_i)$ denotes the neighborhood of the node v_i.

Definition 5.2.3. Distinguishing Set for v_i and $v_j = DS(v_i, v_j) = CN(v_i) \bigoplus CN(v_j)$. $\bigoplus$ denotes the symmetric difference operation between the closed neighborhood sets $CN(v_i)$ and $CNv(j)$. In other words, picking at least one element from the set $DS(v_i, v_j)$ will distinguish between nodes v_i and v_j.

Definition 5.2.4. Universal Set $\mathcal{U} = \{v_1, ..., v_n\}$, where each element v_i is a node in the social network graph and Collection Set $\mathcal{S} = \cup_{i=1}^{n}[CN(v_i) \ \cup_{j=1}^{n} \{DS(v_i, v_j)\}]$.

Our objective is to select the minimum number of elements from $\mathcal{U}$, such that all the elements in $\mathcal{S}$ are hit. Hitting all the elements in $\mathcal{S}$ ensures that, (i) all $CN(v_i)$ sets are hit, which in turn ensures that all nodes in the graph are monitored (a detection sensor has been placed in the closed neighborhood of v_i), and (ii) all $DS(v_i, v_j)$ sets are hit, which in turn ensures that all the nodes in the graph are *uniquely monitored*. Thus the computation of this variant of the minimum HS problem is equivalent to solving the MICS problem.

5.3 Problem Solution

Here, we provide (i) optimal solution for the MICS problem utilizing an Integer Linear Program, based on the HS approach, (ii) heuristic solutions for the MICS problem, by relaxing the integrality constraints of the HS ILP, (iii) an approximation algorithm for the MICS problem with guaranteed performance bound, by utilizing the greedy HS approximation algorithm, and (iv) two minimal heuristics.

5.3.1 Optimal Solution

Instance: A universal set $\mathcal{U} = \{u_1, ..., u_n\}$ and a collection set $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_m\}$, where $\forall i, \mathcal{S}_i \subset \mathcal{U}$.

Problem: Find the smallest subset $\mathcal{U}' \subseteq \mathcal{U}$, which intersects or hits every set $\mathcal{S}_i \in \mathcal{S}$, where $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_m\}$.

Corresponding to each $u_i \in \mathcal{U}$, we use a variable x_i,

$$x_i = \begin{cases} 1, & \text{if } x_i \text{ is included in } \mathcal{U}', \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{u_i \in \mathcal{U}} x_i$

Hitting Constraint: $\sum_{u_i \in \mathcal{S}_i} x_i \geq 1$, $\forall \mathcal{S}_i \in \mathcal{S}$

The objective function ensures that a minimum number of elements are selected from $\mathcal{U}$. The Hitting Constraint ensures that all the sets in $\mathcal{S}$, are hit. We design two heuristics from the LP relaxations from the above ILP. Heuristic 1 (or $LP1_{HS}$) is the relaxed solution where we select the highest fractional values in the LP solution, in descending order, till the graph is uniquely monitored. Heuristic 2 (or $LP2_{HS}$) is the relaxed solution where we select indicator variables independently at random, via randomized rounding, till the graph is uniquely monitored. $LP2$ has a guaranteed error bound of (1 - 1/e) $\sim$ 63% of the optimal (Vazirani 2013).

Algorithm 2 Greedy Hitting Set Approximation Algorithm

```
1: Input: $\mathcal{U}$ and $\mathcal{S}$, the Universal and Collection Sets
2: Output: Return smallest set $U'$ that hits every element in $\mathcal{S}$
3: $U' = \emptyset$
4: while $\mathcal{S} \neq \emptyset$ do
5:     $x = argmax_{x \in \mathcal{U}} |\{\mathcal{S}_i \in \mathcal{S} | x \in \mathcal{S}_i\}|$
6:     $\mathcal{S} = \mathcal{S} \setminus \{\mathcal{S}_i \in \mathcal{S} | x \in \mathcal{S}_i\}$
7:     $U' = U' \cup x$
8: end while
9: return $U'$
```

5.3.2 Approximate Solution

We can now utilize the well known greedy approximation algorithm for HS as an approximation algorithm (Algorithm 2) for the MICS problem (Vazirani 2013). The performance bound for our HS approximation algorithm is $O(log\ n^2)$, since we have a quadratic number of sets in the collection set as a function of the number of elements in the universe. That being said, however, $O(log\ n^2) = O(log\ n)$. Thus, our HS based approach has a performance bound of $O(log\ n)$ and we can claim that the MICS problem also has an $O(log\ n)$ approximation bound. Prior works have already established the $O(log\ n)$ bound for the MICS approximation, utilizing minimizing entropy and computation of disjoint sets (Gravier, Klasing, and Moncel 2008). However, we believe that our transformation is much simpler and easier to implement.

5.3.3 Minimal Solution

In order to make our approach scalable compared to the ILP and approximation algorithm, we present two minimal Identifying Code approaches. We hypothesize that sacrificing a little on the optimality will lead to greater benefits on the computational side. Our minimal algorithms takes as input the graph and a graph centrality based

Algorithm 3 Minimal Algorithm

```
1: Input: $G = (V, E)$, and a node sequence $S$
2: Output: Return the minimal Identifying Code $C$ of $G$
3: $C = S$
4: Place sensors on all nodes in $C$
5: while $S \neq \emptyset$ do
6:     node ← Select the first node in $S$
7:     $NewC = C \setminus \{node\}$
8:     Place sensors on all nodes in $NewC$
9:     if Each node in $G$ is uniquely identifiable then
10:        $C = NewC$
11:    else
12:        $C = C$
13:    end if
14:    Remove node from $S$
15: end while
16: return $C$
```

node sequence. Here, we consider two types of node sequences - (i) nodes arranged in decreasing centrality scores, which we refer to as the MAX approach and, (ii) nodes arranged in increasing centrality scores, which we refer to as the MIN approach. One minimal algorithm takes the graph and MAX node sequence as input and the other takes the graph and MIN node sequence as input. The overall generalized minimal algorithm is presented in Algorithm 3. Further, we have considered four standard graph centrality measures - degree (DC), betweenness (BC), eigenvector (EC) and pagerank (PR) for comparison in the experimental section.

5.4 Experimental Results

We implemented our approaches on various real world *undirected social network* datasets obtained from (Leskovec and Krevl 2014; Rossi and Ahmed 2016). For a network to have an Identifying Code, it must be "twin-free" and, one trivial Identifying Code set solution is the set V itself, although, V may not be the Identifying Code set of *minimum/minimal cardinality.* However, our algorithms show that unique

Network ID	Num. Nodes	Num. Edges	Num. Nodes Post Twin Removal	Num. Edges Post Twin Removal	ILP HS	LP1 HS	LP2 HS	HS APP	MAX DC	MAX BC	MAX EC	MAX PR	MIN DC	MIN BC	MIN EC	MIN PR
FB1	52	198	46	181	18	23	25	19	23	25	23	24	18	18	18	18
FB2	61	331	56	299	18	31	23	26	25	26	25	26	24	24	24	24
FB3	150	1843	144	1731	32	46	45	34	46	47	45	45	38	36	38	38
FB4	168	1824	166	1817	30	53	49	32	60	57	52	57	32	32	33	32
FB5	224	3416	220	3393	40	63	59	44	68	67	62	69	46	44	46	47
FB6	333	2852	312	2730	85	119	128	93	125	124	120	128	88	89	90	88
FB7	534	5347	517	5203	113	167	206	127	174	179	169	177	126	130	134	125
FB8	747	30772	744	30756	82	149	157	95	139	140	125	146	114	102	113	112
FB9	786	14810	767	14702	105	176	188	120	196	204	190	204	112	123	127	120
FB10	1034	27783	1026	27742	126	234	237	143	238	232	217	241	153	148	160	153
FB11	4039	92273	3951	91577	–	–	–	766	1128	1143	1081	1150	788	787	810	789
FB12	6386	224048	6374	224014	–	–	–	–	1560	1593	1471	1580	764	748	779	752
FB13	8600	393126	8590	393102	–	–	–	–	1837	1863	1729	1888	839	811	853	826
FB14	11247	362605	11245	362601	–	–	–	–	2789	2820	2604	2844	1274	1232	1300	1257
FB15	18448	992366	18448	992366	–	–	–	–	3763	3826	3512	3832	1525	1470	1592	1506
FB16	22900	875319	22894	875295	–	–	–	–	5629	5699	5255	5763	2603	–	2648	2569
FB17	27737	1062539	27730	1062518	–	–	–	–	6566	–	6137	6680	3016	–	3108	2968
FB18	29747	1335512	29738	1335487	–	–	–	–	6448	–	5977	5977	2801	–	2884	2757
FB19	32375	1151149	32361	1151102	–	–	–	–	7659	–	6819	7788	3476	–	3565	3397

Table 21. Minimum (Minimal) detection sensors required. – indicates that the algorithm did not finish computation within a specific time frame. The Best performing MAX and MIN approach has been marked in Blue and Purple respectively.

identification for all the nodes in the network can be obtained by monitoring a subset $V' \subseteq V$.

5.4.1 Datasets

The instances of the Facebook networks varied from 52 nodes to 32375 nodes, and are denoted as FB1-FB19 in Table 21. However, due to computational limitations of the laptop on which our methods were implemented, we were not able to consider graphs larger than 32000 nodes. Observe that, almost all of the networks initially contained "twins" in Table 21. For instance, FB1 had 52 nodes initially and after a simple "twin" removal procedure, where "twins" were condensed into a single node (super node), was left with 46 "twin" free nodes. Since "twins" were condensed into a single node, if the condensed node were to become active in misinformation dissemination, then additional lower level analysis would be required to distinguish between the "twins". Table 21 presents the various datasets considered for our experimentation and corresponding network statistics with results.

5.4.2 Analyses

Our objective was two-fold - (i) to show that Integer Linear Programs cannot be utilized for unique coverage (unique monitoring) apart from fairly small problem instances, and (ii) the minimal algorithm, which we presented in this chapter, not only scales well, but also provides similar quality solutions as that of the ILPs. All the experiments were executed on a 5th generation Intel Core i-5 processor with 2.30 GHz and 64GB RAM. We present the results of our analyses in Table 21 and as plots, illustrated in Figure 18 - Figure 21.

We present the number of sensors required to uniquely monitor the corresponding social networks in Table 21[4]. The first column in the table indicates the unique ID of the network. The following two columns report the original number of nodes and edges in the social network (graph), before "twin" removal. The subsequent two columns records the number of nodes and edges post "twin" removal. The next columns indicate the minimum number of sensors required following the various approaches outlined previously. Note that the ILPs and its linear relaxations did not finish computing the number of sensors required for networks FB11 - FB19, within 12 hours of CPU clock time. The HS Approximation algorithm also, could not finish the computation of the number of sensors required for networks FB12 - FB19, within 16 hours. Finally, the MAX BC and MIN BC approaches could not finish the computations for FB17-FB19 and FB16-FB19 respectively, within 16 hours, as the worst case computation of the

[4]Best Viewed In Color

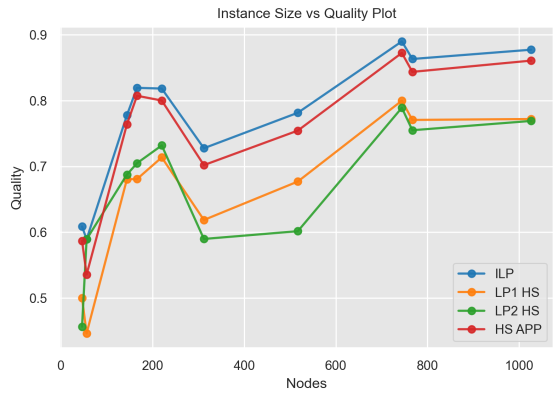
betweenness centrality is $\sim O(n^2)$ time (Brandes 2001), where n denotes the number of nodes. The computation times of 12 and 16 hours were determined based on the computation times of the other approaches on the corresponding graphs. All of the unfinished computations are denoted by $-$ in Table 21.

In our experimentation, we use two notions extensively, *quality* and *cost*. We define *quality* as the "goodness" of a solution, which can be mathematically represented as,

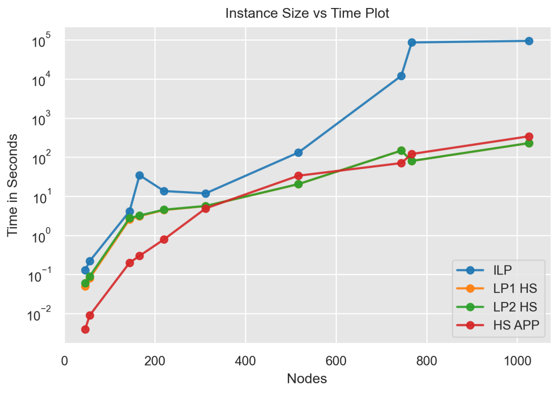
$$Q = (Num.Nodes - Solution\ Cardinality)/Num.Nodes \tag{5.1}$$

The value of Q lies in the interval $[0, 1]$, where 1 represents a high quality solution, or in other words, a solution with high reduction in resources. Lower the value of the solution cardinality, higher the value of Q and higher is the quality of an approach. Next, we define *cost* as the computational time spent in order to attain the respective quality. Figure 18a[5] illustrates the quality of the ILP, LPs and approximation approaches outlined previously, for FB1-FB10. This is because, FB10 is the largest instance which we could provide as input to the ILP. Hence, for initial comparison, we plot the quality and cost results of the ILPs, LPs and HS approximation as obtained from FB1-FB10. As expected, the quality of the ILP is the best. In this figure, we can see that our *Hitting Set (HS) approximation approach produces near optimal solution quality.* The qualities of the linear relaxations of the ILP are also presented in the figure, but are not as high as the quality of the approximation algorithm. Figure 18b illustrates the cost associated with the approaches for obtaining the corresponding quality. It should be observed that the y-axis in Figure 18b is in the log-scale because - (i) the quality of the smaller networks (FB1-FB2) was computed in less than a second, and (ii) to make the plot more visually more understandable. The cost of the ILP

[5]Images Best Viewed in Color



(a) Quality Analysis

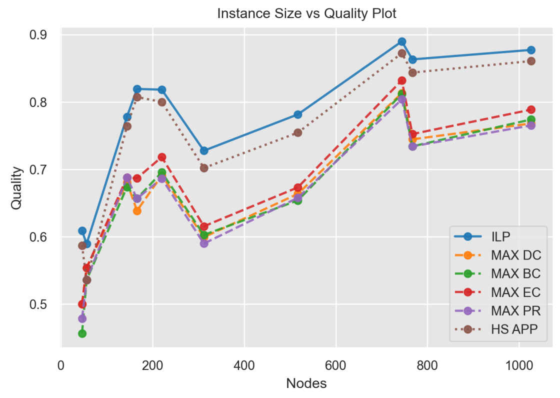


(b) Cost Analysis

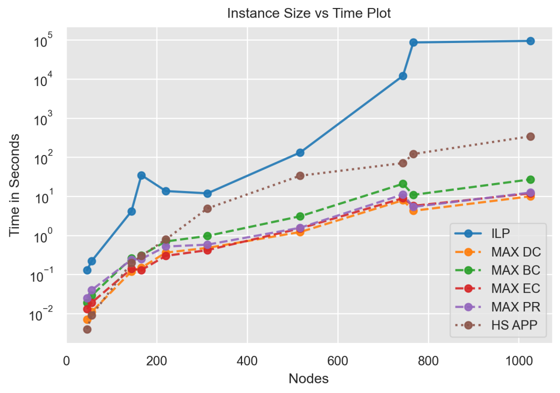
Figure 18. Visual Analysis of The ILP, LPs and Approximation Performances For FB1-FB10

approach tends to grow exponentially with an increase in the problem instance. The growth curves of the other approaches, including the approximation algorithm, are not as extreme as that of the ILP. There are a few takeaways from these two plots, (i) the cost of the approximation algorithm, while producing near optimal solution, also tends to grow exponentially with an increase in the size of the problem instance (a linear growth in log-scale implies exponential growth in normal scale), (ii) it is due to such growth in the cost that, we could only implement these approaches on small graphs/networks, more specifically FB1-FB10.

Having established the fact that the optimal approach and our approximation approach do not scale well, we sacrifice on the objective of attaining the minimum (or even approximate) solution and settle for the *minimal solution*, and present the results of our analyses utilizing the minimal algorithm (with its two node sequence orderings). It may be recalled that we considered two groups of node sequence orderings based on

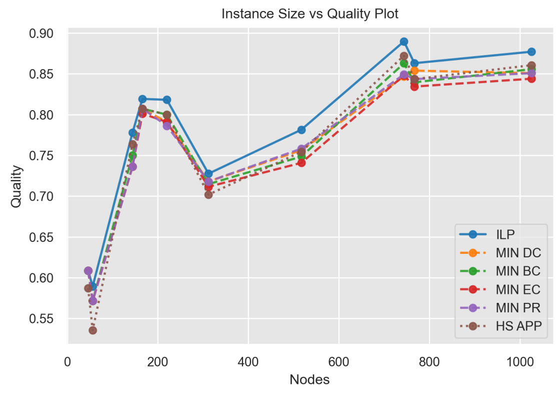


(a) Quality Analysis

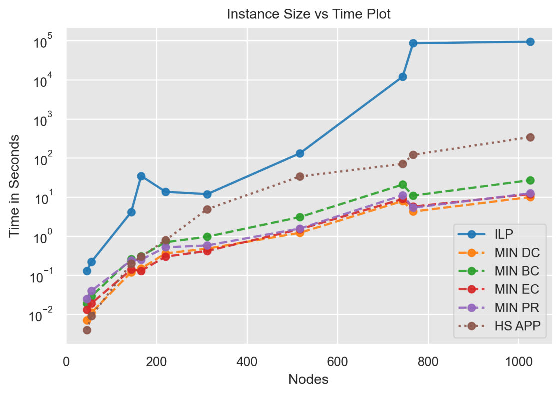


(b) Cost Analysis

Figure 19. Visual Analysis of The ILP, Approximation and Minimal Algorithm (MAX) Performances For FB1-FB10



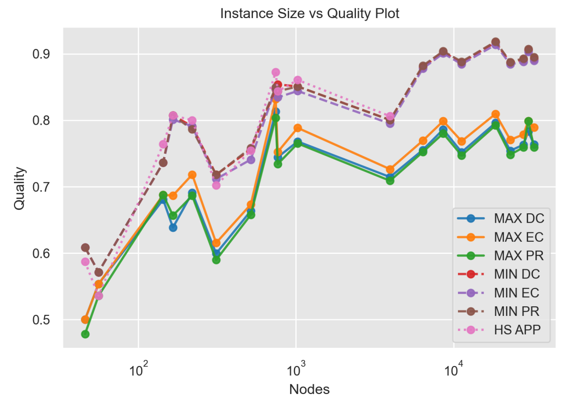
(a) Quality Analysis



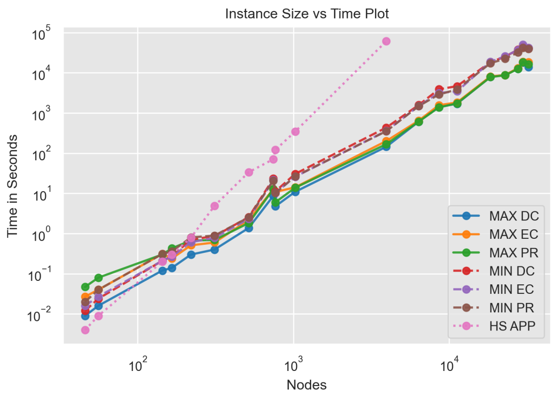
(b) Cost Analysis

Figure 20. Visual Analysis of The ILP, Approximation and Minimal Algorithm (MIN) Performances For FB1-FB10

common centrality measures, the maximum ordering with MAX DC, MAX BC, MAX EC and MAX PR, and the minimum ordering with MIN DC, MIN BC, MIN EC and MIN PR. *To show the efficacy of our minimal approach, we compare the quality and cost of each variation with the ILPs and approximation algorithm for smaller networks, before moving on to larger networks.* Figure 19 illustrates the comparison of the qualities and costs of MAX DC, MAX BC, MAX EC and MAX PR with the quality and cost of the ILPs and approximation algorithm respectively, for FB1- FB10.



(a) Quality Analysis



(b) Cost Analysis

Figure 21. Visual Analysis of The Approximation and The Two Minimal Approaches For FB1-FB19

Figure 19 illustrates that the maximum ordering attains comparatively high quality (greater than 0.65 on average) with MAX EC providing the best quality. Our minimal approach, based on MAX ordering, approximately attains at least 10x scalability, when compared to the approximation algorithm, as the sizes of the networks keep increasing. It can also be observed that the MAX BC approach is the most expensive approach, among the four maximum orderings, whereas the MAX DC approach is the least expensive. Overall, in terms of quality, the approaches MAX EC and MAX DC do not vary much, but in terms of cost, the MAX DC is more efficient than the MAX EC approach. Figure 20 illustrates the comparison of the qualities and costs of MIN DC, MIN BC, MIN EC and MIN PR with the quality and cost of the ILPs and approximation algorithm respectively, for FB1 - FB10. Figure 20 illustrates that the MIN ordering attains almost the same quality, if not better, when compared to the approximation algorithm (averaging greater than 0.75). Among the minimum orderings, MIN BC provides the best quality, whereas the MIN PR and MIN DC are the most efficient. Overall, the difference in cost between the MIN BC and MIN PR /

MIN DC is significantly larger than difference in quality between the two, and hence, one may opt for the more efficient MIN PR/ MIN DC approach.

Now, we compare the performances of the minimal algorithm following the MAX ordering sequence with those following the MIN ordering sequence. Note that, MAX BC and MIN BC failed to execute on larger graphs, and hence, we do not consider these two approaches going forward. We include the HS Approximation algorithm for instances it managed to finish its execution (FB1-FB11). Figure 21 illustrates the quality and cost performances of the minimal and approximation algorithms. In Figure 21, it can be seen that the quality of the minimal algorithms following the MIN ordering sequence is almost identical to that of the HS Approximation algorithm and the quality of the minimal algorithms following the MAX ordering sequence is slightly lower. In Figure 21, we see that both the minimal approaches grow at a smaller rate as compared to the HS Approximation algorithm. In fact, both the minimal approaches are at least 10x faster than the approximation algorithm. As the size of the instances increases, this gap widens to almost 100x. The minimal algorithms following the MAX ordering are computationally more efficient than those of the MIN ordering.

It can be observed that the quality of the minimal approach following the MIN ordering is as good as the quality of the approximation algorithm, if not better, while bearing only a fraction of the cost. Moreover, the quality achieved by the minimal algorithm following MIN ordering is far superior to that achieved by MAX ordering as, in the case of MAX ordering, nodes removed from consideration (i.e., step 5 of Algorithm 3) are the nodes with higher centrality scores. If such central or important nodes are not considered for sensor placement, then the *effect* of reach of that highly central node is lost, in the sense that, placing a monitor at a more central node would monitor *more* nodes than that of placing a monitor at a lesser central node. Thus, removing highly central nodes from consideration at every iteration would result in the selection of additional nodes to make up for this loss, resulting in the higher solution cardinality. This is not the case for the MIN ordering, where nodes with low scores are removed from consideration first, thus resulting in a smaller loss in effect as opposed to the MAX ordering based approach. Thus, the cost associated with the MIN ordering is higher as a larger set of nodes are being removed from consideration when compared to the MAX ordering.

Following our experimentation, it can be seen that MIN DC, MIN EC and MIN PR all provide near identical quality and cost. MAX EC provides the best quality and has equivalent cost to the other MAX approaches. A key point which we want to make here is that, all of these experiments were conducted on a laptop. We believe that with better computing resources, our MIN and MAX ordering based minimal algorithms could be scaled even further.

5.5 Conclusion and Future Research Directions

In this chapter, we have shown how Identifying Code trumps prior studies which focused on an underlying epidemiological model. Firstly, our framework does not assume any underlying epidemiological model for dissemination and secondly, does not know in advance, the set of users (nodes) who have been already infected with misinformation. Another major advantage of our framework is the universality - by changing the objective of the detection sensor, one can monitor any objectionable content propagating on an online social network (for e.g., cyber bullying, hate speeches, discriminations, etc.). Moreover, an approximation algorithm with guaranteed performance bound was utilized, along with a minimal algorithm with varying input node sequences. We compared our approaches with ILPs and showed that the ILPs as well as their linear relaxations and the approximation algorithm, do not scale particularly well. However, we showed that our minimal algorithms not only scaled well, but also provided solution quality almost as good as the approximation algorithm, if not better, while only being fractionally costly.

It may be argued that our approach needs to be implemented every time the graph changes. This is not necessarily true as one might incorporate a simple augmenting approach to handle cases where new users join the platform, similar to the approach discussed in Chapter 3. Since the locations of the sensor deployment are known, we can easily check if the new node(s) will be uniquely monitored and accordingly, we can monitor the appropriate node(s). Since the number of new nodes joining the platform will be significantly lesser than those who are already present, this augmenting task will not be computationally expensive. Finally, it may be noted that our approach is not robust (unique identification is lost due to sensor failures or adversarial attacks). That being said however, we have identified certain future directions for this domain:

- *Bot Identification*: Nowadays, bots are rampant on social networks and spread curated propaganda at a high volume. The approach presented in this chapter does not deal with the presence of bots in the social network. For the analysis of a realistic scenario, the approach must be modified in order to accurately and quickly identify these bots before falsehoods can be propagated to a large part of the network.
- *Examining Lagrangian Relaxation*: The minimal approach presented in this chapter has achieved a certain scalability factor when compared to the approximation algorithm and the ILP. Even then, the graph instances studied are no where close to the actual sizes of a social network graph. Consequently, it could

be of note to examine the sophisticated Lagrangian relaxation technique, which have been used in other studies to attain appreciable factors of scalability.

Chapter 6

WATER DISTRIBUTION NETWORKS

Water Distribution Networks (WDNs) form one of the many critical infrastructure of a modern day city. Such networks are built to purify, regulate and supply a large metropolitan area with water. However, it has been seen over the past couple of decades that such networks need to be continuously monitored. For instance, a spread of a contaminant in Milwaukee's WDN affected more than 400,000 people. The cause was later determined to be a microorganism which was transported throughout the system. Around 8500 were taken ill after a cross connection of the WDN with wastewater occurred in Nokia, Finland (Kauppinen et al. 2019). Consequently, it has become critical to monitor the WDNs for any contaminants and/or leaks (anomalous behavior). In this chapter, inspired by the problem studied at The Battle of the Water Sensor Networks (BWSN, Ostfeld et al. 2008), we present a novel variant of the Identifying Code problem which overcomes a limitation of the approach presented at BWSN. A modified Max k-Cover approach was designed as a solution technique during BWSN. We show, in this chapter, that our novel Budget Constrained Identifying Code Set (BCICS) problem can be utilized to overcome this limitation.

Sensors are used extensively for monitoring various parameters so that any anomalous behaviour can easily be detected (Noel et al. 2017; Bhuiyan et al. 2014; Basu et al. 2019; Krause et al. 2008). Sensors in a deployment area have two functions, (i) sensing/coverage of target parameters such as temperature, pressure, vibration, etc., and (ii) to transmit the sensed data either directly or through multiple other sensor nodes (which serves as relays) to the control station for the analysis of the sensed data. Over the years several coverage models have been proposed, where the underlying assumption is that a sensor placed in a certain location, can sense its environment up to a certain distance. In other words, the sensors have a specific *sensing range* associated with it. This assumption often leads to a Set Cover based problem formulation (Wang 2011; Liang, Shen, and Chen 2021), which unfortunately has a serious limitation, in the sense that it lacks unique identification capability for the location where anomalous behavior is sensed. We have elaborated this limitation in detail in Chapter 2. This point can be summarized in the following way. Assume that a set of sensors, $\mathcal{S} = \{S_1, ..., S_n\}$, is placed in a deployment area to monitor a set of *Points-of-Interest*, $P = \{p_1, ..., p_m\}$. Suppose that a subset $P' \subseteq P$ is within the sensing range of a specific sensor $S_i, S_i \in \mathcal{S}$. In this situation, if any point $p_j \in P'$ starts behaving anomalously, this behavior will be registered by sensor S_i. However, S_i will not be able to determine whether the misbehaving device is at point p_j or any other point p_k, as long as p_k is also within the sensing range of S_i, i.e., $p_k \in P'$. This limitation also exists in the max k-cover problem, a variation of Set Cover.

This limitation can be overcome through utilization of Identifying Code (IC). The optimal solution of the IC problem provides the minimum number of sensors that will be needed to uniquely identify the location where anomalous behavior is sensed. In this chapter, we introduce a *budget constrained* version of the problem, whose goal is to find the largest number of locations that can be uniquely identified with the sensors that can be deployed *within the specified budget*. To the best of our knowledge, the budget constrained version of the IC problem has not been studied before. We provide an Integer Linear Programming formulation and a Maximum Set-Group Cover (MSGC) formulation (a generalized version of the Max k-Cover) for the problem. We prove that the MSGC problem cannot have a polynomial time approximation algorithm with a 1/k factor performance guarantee unless P = NP. Our experiments focuses on detecting sources of contaminants in Water Distribution Networks (WDNs). We chose WDNs because of (i) its obvious importance as a critical infrastructure to smart cities; (ii) it has been extensively studied by other researchers (Eliades and Polycarpou 2009; Krause et al. 2008; Hart and Murray 2010; Lee and Deininger 1992); and (iii) synthetic and real WDN graphs are readily available in the public domain (Kentucky 2001) for conducting experiments. Prior works have primarily utilized budgeted sensor placement approaches (max k-cover) for the detection of contaminants in such networks (Krause et al. 2008). In this chapter, we take this one step further and propose an approach which can not only detect contaminants but also identify its sources, utilizing the concept of ICs.

6.1 Related Work

The coverage aspect of sensor networks alone has been studied extensively (Wang 2011; Cardei and Wu 2004). The survey on Coverage Problems in Sensor Networks (Wang 2011), references close to 200 papers. Thus, a multitude of sensor coverage models, such as (i) Boolean Sector Coverage Model, (ii) Boolean Disc Coverage Model, (iii) Attenuated Disc Coverage Model, (iv) Truncated Attenuated Disc Models, (v) Estimation Coverage Models, etc. have been studied by various research groups (Wang 2011). A more recent survey (Tripathi et al. 2018) lists additional studies on the topic. Cardei and Wu in (Cardei and Wu 2004), classified coverage problems into three broad classes, (i) Point Coverage, (ii) Area Coverage, and (iii) Barrier Coverage. While in the area coverage problem, an entire area (in two or three dimensional space) has to be sensed (monitored), in the point coverage problem, only a *specified set of points* (points of interest) in 2D or 3D space, has to be monitored. Oftentimes, there are restrictions on the locations (in 2D/3D space), where the sensors can be deployed (cost, terrain, natural elements, etc.), thereby reducing the number of available placement locations. For example, in case of earthquake monitoring, regions where some fault line passes, are more important from the points of interest perspective than regions where there are no such fault lines. Similarly, sensors such as seismometers or accelerometers cannot be deployed in any arbitrary location, due to various constraints such as, cost of deployment, access difficulty, property rights, etc. Accordingly, sensor coverage problems can be classified in the following way.

Sensor coverage problems have a deployment area, where the Points of Interest (PoI) are located and the sensors have to be deployed. A deployment area can be thought of as an (infinite) set of points in a two or three dimensional space. If R, P and Q denote the (infinite) set of points in the deployment area, PoIs in the deployment area and potential locations for sensor placement respectively, then four different case scenarios can be considered, (i) $R = P = Q$, (ii) $R = P$ and $Q \subset R$, (iii) $P = Q$ and $P, Q \subset R$, and (iv) $P \neq Q$ and $P, Q \subset R$. The cases 1 and 2 are considered as *Area Coverage problem* whereas cases 3 and 4 are considered as *Point Coverage problem* (according to (Cardei and Wu 2004)), which is the focus of this chapter.

The most frequent studied problem in this context is the Sensor Placement Optimization, whose goal is to find the smallest set of locations to deploy sensors, so that all the points of interest can be monitored. If a Boolean disc coverage model (Wang 2011) is used for sensor coverage, the Sensor Placement Optimization problem can be formulated as a Set Cover problem (Kleinberg and Tardos 2006) and a number of studies using this model are available in the literature (Wang 2011; Cardei and Wu 2004). Sensor placement for the detection of anomalies in smart cities has been previously studied in (Hancke, Hancke Jr, et al. 2013; Quadar et al. 2021), where the latter work describes technologies utilized in sensing locations of interest in smart cities (with a case study on a water distribution network).

Identifying Codes (IC) was introduced by Karpovsky *et al.* in (Karpovsky, Chakrabarty, and Levitin 1998) and other researchers have followed up by studying it both from a theoretical and applicative perspective (Foucaud 2015; Ray et al. 2003). Laifenfeld *et al.* studied joint monitoring and routing in wireless sensor networks with IC in (Laifenfeld et al. 2009). Ray *et al.* studied location detection problem in emergency sensor networks, using IC (Ray et al. 2003) and presented an algorithm for generating *irreducible* IC in polynomial time. Note that irreducible IC is only a *minimal* IC and may not be the *minimum* (or optimal) IC. Basu *et al.* presented Integer Linear Programs for the computation of the minimum IC set in (Basu, Zhou, et al. 2018; Basu, Padhee, et al. 2018; Basu and Sen 2019a, 2019b, 2021b; Arunabha Sen et al. 2018) for problems arising from multiple domains, varying from drug/terrorist network monitoring to monitoring critical infrastructures. Sengupta *et al.* utilized Moving Target Defense and IC to present an approach which prevented cyber attacks on sensors placed in critical infrastructures in (Sengupta, Basu, et al. 2020). The problem of identifying misinformation spreaders in social networks was studied in (Basu 2019; Basu and Sen 2021a). Padhee *et al.* utilized ICs for identifying vulnerable assets in critical power systems in (Padhee et al. 2020).

While the goal of all these studies was to find the *minimum* number of sensors to monitor all PoIs, we consider a *budget constrained version*, where the number of sensors that can be deployed is *limited by a pre-approved budget*. Our aim is to *maximize* the number of PoIs that can be monitored with the limited number of sensors that can be procured and deployed. Moreover, we show how our approach can help with the monitoring of Smart Cities, as technologies continue to develop for the same (Anthopoulos 2017).

6.2 From Points on the Plane to Graphs

Previously, we described the sensor placement problem in terms of two sets of points in a plane, whereas the Minimum Identifying Code Set (MICS) problem is described in terms of a graph. From the set of points $\mathcal{S}$ and P, we can construct a graph using the following construction rules: (i) For each point $S_i \in \mathcal{S}$ and each point $p_i \in P$, we create node in the graph. Thus the node set V in the $G = (V, E)$, is $V_{\mathcal{S}} \cup V_P$. If a point $p_i \in P$ is within the sensing range of a sensor S_j, then there is an edge in the graph connecting the nodes v_{p_i} and v_{S_j}, where v_{p_i} and v_{S_j} are the nodes corresponding to the points p_i and S_j respectively.

Budget Constrained Identifying Code Set (BCICS) Problem: Given a graph $G = (V, E)$ and an integer B, a subset $V' \subseteq V$ of cardinality at most B is called the Budget Constrained Identifying Code Set of $G = (V, E)$, if it *maximizes* $|V''|$, where $V'' \subseteq V$ and for no two nodes v_i, v_j, such that $v_i \in V''$ and $v_j \in V$, $N^+(v_i) \cap V' \neq N^+(v_j) \cap V'$.

It may be recalled that the main objective of this study is to deploy the sensors that can be procured within the specified budget, as judiciously as possible, so as to *maximize* the number of locations (points of interest) that can have unique fault identification signature. Moreover, in this study we focus our attention only to the scenario where *abnormality (or failure) is restricted to only one point.* As mentioned previously, in this chapter, we do not consider multiple simultaneous failure.

6.3 Solutions for BCICS Problem

In this section, we first show that BCICS can be set up as a generalization of the well studied *Maximum Set Cover* (MSC) problem (Khuller, Moss, and Naor 1999), referred in this chapter as *Maximum Set-Group Cover* (MSGC) problem. Next, we present an optimal solution to the BCICS problem.

6.3.1 Maximum Set-Group Cover Formulation of BCICS Problem

For convenience, we first state the MSC and then the MSGC problems. Then, we explain with the help of an example how the generalization of MSC to MSGC can be used to solve the BCICS problem.

$a_1 = CN(1) = \{1,5,6,7\}$	$a_2 = CN(2) = \{2,5,8,9\}$	$a_3 = CN(3) = \{3,6,8,10\}$	$a_4 = CN(4) = \{4,7,9,10\}$
$a_5 = CN(5) = \{1,2,5\}$	$a_6 = CN(6) = \{1,3,6\}$	$a_7 = CN(7) = \{1,4,7\}$	$a_8 = CN(8) = \{2,3,8\}$
$a_9 = CN(9) = \{2,4,9\}$	$a_{10} = CN(10) = \{3,4,10\}$	$a_{11} = DS(1,2) = \{1,2,6,7,8,9\}$	$a_{12} = DS(1,3) = \{1,3,5,7,8,10\}$
$a_{13} = DS(1,4) = \{1,4,5,6,9,10\}$	$a_{14} = DS(1,5) = \{2,6,7\}$	$a_{15} = DS(1,6) = \{3,5,7\}$	$a_{16} = DS(1,7) = \{4,5,6\}$
$a_{17} = DS(1,8) = \{1,2,3,5,6,7,8\}$	$a_{18} = DS(1,9) = \{1,2,4,5,6,7,9\}$	$a_{19} = DS(1,10) = \{1,3,4,5,6,7,10\}$	$a_{20} = DS(2,3) = \{2,3,5,6,9,10\}$
$a_{21} = DS(2,4) = \{2,4,5,7,8,10\}$	$a_{22} = DS(2,5) = \{1,8,9\}$	$a_{23} = DS(2,6) = \{1,2,3,5,6,8,9\}$	$a_{24} = DS(2,7) = \{1,2,4,5,7,8,9\}$
$a_{25} = DS(2,8) = \{3,5,9\}$	$a_{26} = DS(2,9) = \{4,5,8\}$	$a_{27} = DS(2,10) = \{2,3,4,5,8,9,10\}$	$a_{28} = DS(3,4) = \{3,4,6,7,8,9\}$
$a_{29} = DS(3,5) = \{1,2,3,5,6,8,10\}$	$a_{30} = DS(3,6) = \{1,8,10\}$	$a_{31} = DS(3,7) = \{1,3,4,6,7,8,10\}$	$a_{32} = DS(3,8) = \{10,2,6\}$
$a_{33} = DS(3,9) = \{2,3,4,6,8,9,10\}$	$a_{34} = DS(3,10) = \{4,6,8\}$	$a_{35} = DS(4,5) = \{1,2,4,5,7,9,10\}$	$a_{36} = DS(4,6) = \{1,3,4,6,7,9,10\}$
$a_{37} = DS(4,7) = \{1,9,10\}$	$a_{38} = DS(4,8) = \{2,3,4,7,8,9,10\}$	$a_{39} = DS(4,9) = \{2,7,10\}$	$a_{40} = DS(4,10) = \{3,7,9\}$
$a_{41} = DS(5,6) = \{2,3,5,6\}$	$a_{42} = DS(5,7) = \{2,4,5,7\}$	$a_{43} = DS(5,8) = \{1,3,5,8\}$	$a_{44} = DS(5,9) = \{1,4,5,9\}$
$a_{45} = DS(5,10) = \{1,2,3,4,5,10\}$	$a_{46} = DS(6,7) = \{3,4,6,7\}$	$a_{47} = DS(6,8) = \{1,2,6,8\}$	$a_{48} = DS(6,9) = \{1,2,3,4,6,9\}$
$a_{49} = DS(6,10) = \{1,4,6,10\}$	$a_{50} = DS(7,8) = \{1,2,3,4,7,8\}$	$a_{51} = DS(7,9) = \{1,2,7,9\}$	$a_{52} = DS(7,10) = \{1,3,7,10\}$
$a_{53} = DS(8,9) = \{3,4,8,9\}$	$a_{54} = DS(8,10) = \{2,4,8,10\}$	$a_{55} = DS(9,10) = \{2,3,9,10\}$	

Table 22. $CN(v_i)$ and $DS(v_i, v_j)$ Table for all $i, j, \quad 1 \le i, j \le n$; $A = \{a_1, \ldots, a_{55}\}$

$A'_1 = PS(1) = \{a_1, a_5, a_6, a_7, a_{11}, a_{12}, a_{13}, a_{17}, a_{18}, a_{19}, a_{22}, a_{23}, a_{24}, a_{29}, a_{30}, a_{31}, a_{35}, a_{36}, a_{37}, a_{43}, a_{44}, a_{45}, a_{47}, a_{48}, a_{49}, a_{50}, a_{51}, a_{52}\}$
$A'_2 = PS(2) = \{a_2, a_5, a_8, a_9, a_{11}, a_{14}, a_{17}, a_{18}, a_{20}, a_{21}, a_{23}, a_{24}, a_{27}, a_{29}, a_{32}, a_{33}, a_{35}, a_{38}, a_{39}, a_{41}, a_{42}, a_{45}, a_{47}, a_{48}, a_{50}, a_{51}, a_{54}, a_{55}\}$
$A'_3 = PS(3) = \{a_3, a_6, a_8, a_{10}, a_{12}, a_{15}, a_{17}, a_{19}, a_{20}, a_{23}, a_{25}, a_{27}, a_{28}, a_{29}, a_{31}, a_{33}, a_{36}, a_{38}, a_{40}, a_{41}, a_{43}, a_{45}, a_{46}, a_{48}, a_{50}, a_{52}, a_{53}, a_{55}\}$
$A'_4 = PS(4) = \{a_4, a_7, a_9, a_{10}, a_{13}, a_{16}, a_{18}, a_{19}, a_{21}, a_{24}, a_{26}, a_{27}, a_{28}, a_{31}, a_{33}, a_{34}, a_{35}, a_{36}, a_{38}, a_{42}, a_{44}, a_{45}, a_{46}, a_{48}, a_{49}, a_{50}, a_{53}, a_{54}\}$
$A'_5 = PS(5) = \{a_1, a_2, a_5, a_{12}, a_{13}, a_{15}, a_{16}, a_{17}, a_{18}, a_{19}, a_{20}, a_{21}, a_{23}, a_{24}, a_{25}, a_{26}, a_{27}, a_{29}, a_{35}, a_{41}, a_{42}, a_{43}, a_{44}, a_{45}\}$
$A'_6 = PS(6) = \{a_1, a_3, a_6, a_{11}, a_{13}, a_{14}, a_{16}, a_{17}, a_{18}, a_{19}, a_{20}, a_{23}, a_{28}, a_{29}, a_{31}, a_{32}, a_{33}, a_{34}, a_{36}, a_{41}, a_{46}, a_{47}, a_{48}, a_{49}\}$
$A'_7 = PS(7) = \{a_1, a_4, a_7, a_{11}, a_{12}, a_{14}, a_{15}, a_{17}, a_{18}, a_{19}, a_{21}, a_{24}, a_{28}, a_{31}, a_{35}, a_{36}, a_{38}, a_{39}, a_{40}, a_{42}, a_{46}, a_{50}, a_{51}, a_{52}\}$
$A'_8 = PS(8) = \{a_2, a_3, a_8, a_{11}, a_{12}, a_{17}, a_{21}, a_{22}, a_{23}, a_{24}, a_{26}, a_{27}, a_{28}, a_{29}, a_{30}, a_{31}, a_{33}, a_{34}, a_{38}, a_{43}, a_{47}, a_{50}, a_{53}, a_{54}\}$
$A'_9 = PS(9) = \{a_2, a_4, a_9, a_{11}, a_{13}, a_{18}, a_{20}, a_{22}, a_{23}, a_{24}, a_{25}, a_{27}, a_{28}, a_{33}, a_{35}, a_{36}, a_{37}, a_{38}, a_{40}, a_{44}, a_{48}, a_{51}, a_{53}, a_{55}\}$
$A'_{10} = PS(10) = \{a_3, a_4, a_{10}, a_{12}, a_{13}, a_{19}, a_{20}, a_{21}, a_{27}, a_{29}, a_{30}, a_{31}, a_{32}, a_{33}, a_{35}, a_{36}, a_{37}, a_{38}, a_{39}, a_{45}, a_{49}, a_{52}, a_{54}, a_{55}\}$

Table 23. $PS(v_i)$ Table for all $i, \quad 1 \le i \le n$

$G_1 = IS(1) = \{a_1, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{16}, a_{17}, a_{18}, a_{19}\}$
$G_2 = IS(2) = \{a_2, a_{11}, a_{20}, a_{21}, a_{22}, a_{23}, a_{24}, a_{25}, a_{26}, a_{27}\}$
$G_3 = IS(3) = \{a_3, a_{12}, a_{20}, a_{28}, a_{29}, a_{30}, a_{31}, a_{32}, a_{33}, a_{34}\}$
$G_4 = IS(4) = \{a_4, a_{13}, a_{21}, a_{28}, a_{35}, a_{36}, a_{37}, a_{38}, a_{39}, a_{40}\}$
$G_5 = IS(5) = \{a_5, a_{14}, a_{22}, a_{29}, a_{35}, a_{41}, a_{42}, a_{43}, a_{44}, a_{45}\}$
$G_6 = IS(6) = \{a_6, a_{15}, a_{23}, a_{30}, a_{36}, a_{41}, a_{46}, a_{47}, a_{48}, a_{49}\}$
$G_7 = IS(7) = \{a_7, a_{16}, a_{24}, a_{31}, a_{37}, a_{42}, a_{46}, a_{50}, a_{51}, a_{52}\}$
$G_8 = IS(8) = \{a_8, a_{17}, a_{25}, a_{32}, a_{38}, a_{43}, a_{47}, a_{50}, a_{53}, a_{54}\}$
$G_9 = IS(9) = \{a_9, a_{18}, a_{26}, a_{33}, a_{39}, a_{44}, a_{48}, a_{51}, a_{53}, a_{55}\}$
$G_{10} = IS(10) = \{a_{10}, a_{19}, a_{27}, a_{34}, a_{40}, a_{45}, a_{49}, a_{52}, a_{54}, a_{55}\}$

Table 24. $IS(v_i)$ Table for all $i, \quad 1 \leq i \leq n$

Definition 6.3.1. Set Cover (SC) Problem: Given a set $A = \{a_1, ..., a_n\}$ and $\mathcal{A}' = \{A'_1, ..., A'_m\}$ $(A'_i \subseteq A, 1 \leq i \leq m)$, find the smallest subset $\mathcal{A}'' \subseteq \mathcal{A}'$ such that all the elements in the set A is covered, i,e., every element of A belong to at least one member of $\mathcal{A}''$.

Definition 6.3.2. Maximum Set Cover (MSC) Problem: Given a set $A = \{a_1, ..., a_n\}$ and subsets $\mathcal{A}' = \{A'_1, ..., A'_m\}$ $(A'_i \subseteq A, 1 \leq i \leq m)$ and an integer B, find the largest subset $A'' \subseteq A$ that can be covered by using a subset $\mathcal{A}''' \subseteq \mathcal{A}'$, where $|\mathcal{A}'''| \leq B$.

Definition 6.3.3. Maximum Set-Group Cover (MSGC) Problem: Given a set $A = \{a_1, ..., a_n\}$ and subsets $\mathcal{A}' = \{A'_1, ..., A'_m\}$ $(A'_i \subseteq A, 1 \leq i \leq m)$ and $\mathcal{G} = \{G_1, ..., G_p\}$ $(G_i \subseteq A, 1 \leq i \leq p)$ and an integer B, find the subset $A'' \subseteq \mathcal{A}'$ with $|A''| = B$ that maximizes the number of *groups* completely "covered" by A'', i.e., it finds the largest cardinality subset $G' \subseteq G$ that satisfies the condition that $\forall G_j \in G'$, $\cup_{A'_i \in A''} A'_i \cap G_j = G_j$.

We elaborate the formulation of the BCICS problem as a MSGC problem with the help of the example shown in Figure 22. We introduce a few definitions before the explanation.

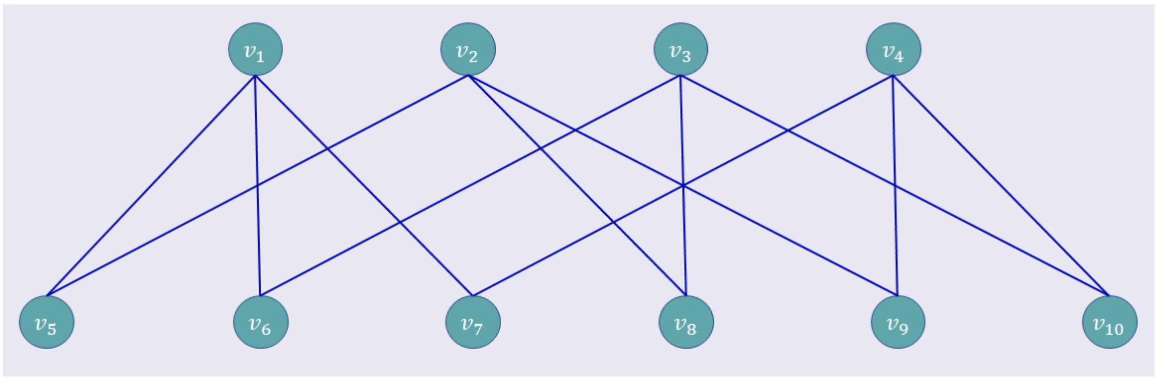


Figure 22. Budgeted Identifying Code Example

Definition 6.3.4. Closed Neighborhood of $v_i = CN(v_i) = N^+(v_i)$, where $N^+(v_i) = N(v_i) \cup \{v\}$, and $N(v_i)$ is the set of nodes adjacent to v_i.

Definition 6.3.5. Distinguishing Set for v_i and $v_j = DS(v_i, v_j) = CN(v_i) \bigoplus CN(v_j)$, where $\bigoplus$ denotes *Exclusive-OR* operation. At least one element of the set must be selected to distinguish between the nodes v_i and v_j.

Definition 6.3.6. Isolation Set for $v_i = IS(v_i) = \cup_{j=1}^{n}\{v_{i_j} : v_{i_j} \in DS(v_i, v_j)\}, j \neq i$. This is the set of sets, such that if all nodes in a set is selected, it will distinguish (isolate/uniquely identify) v_i from all other nodes of the graph.

Definition 6.3.7. Presence Set for v_i, $PS(v_i) = \{CN(v_j) : v_i \in CN(v_j)\} \cup \{DS(v_j, v_k) : v_i \in DS(v_j, v_k)\}, \forall v_i, v_j, v_k, 1 \leq v_i, v_j, v_k \leq n$. $PS(v_i)$ is the set of all $CN(v_j)$s and $DS(v_i, v_j)$s, where v_i is present.

The number of nodes in the graph in Figure 22 is 10, i.e., $n = 10$. Accordingly, there will be 10 $CN(v_i)$ sets and corresponding to each v_i, there will be 9 $DS(v_i, v_j)$ sets, $(\forall v_i, v_j \neq v_i)$. Hence, there will be 100 sets altogether. However, it may be noted that these 100 sets will not be distinct, as $DS(v_i, v_j) = DS(v_j, v_i)$. Thus, the total number of distinct $DS(v_i, v_j)$ sets in this example will be $\sum_{i=1}^{n-1} i = 45$ (as $n = 10$).

These 10 $CN(v_i)$ and 45 $DS(v_i, v_j)$ sets are shown in Table 22, and are marked as a_1 through a_{55}. The Presence Sets for nodes 1 through 10 for the example graph, are shown in Table 23. The Isolation Sets for nodes 1 through 10 for the example graph, are shown in Table 24.

The BCICS problems can be viewed as an MSGC problem in the following way. We say $PS(v_i)$ "hits" $CN(v_j)$ if $PS(v_i) \cap CN(v_j) \neq \emptyset$. Similarly, $PS(v_i)$ "hits" $DS(v_j, v_k)$ if $PS(v_i) \cap DS(v_j, v_k) \neq \emptyset$. With slight misuse of the language, we use the term "cover" instead of "hit", i.e., we will say $PS(v_i)$ "covers" $CN(v_j)$ if $PS(v_i) \cap CN(v_j) \neq \emptyset$ and "covers" $DS(v_j, v_k)$ if $PS(v_i) \cap DS(v_j, v_k) \neq \emptyset$. In Table 22, the $CN(v_j)$ and $DS(v_j, v_k)$ sets are numbered from a_1 through a_{10} and a_{11} through a_{55} respectively ($A = \{a_1, \ldots, a_{55}\}$) . Each $PS(v_i)$ is a subset of the set A, and is denoted as $A'(v_i)$ in Table 23. We define the set $A' = \{A'_1, \ldots, A'_{10}\}$. From Table 22 and Table 23, it can be seen that $A'(1) = PS(1)$ covers $a_1 = CN(1), a_5 = CN(9), a_{37} = DS(4, 7)$ and 25 other $CN(v_i)$ or $DS(v_j, v_k)$ sets shown in the first row of Table 23. The set G is defined as the $IS(v_i), 1 \leq i \leq 10$, i.e., $G = \{G_1, \ldots, G_{10}\}$. Hence, the BCICS problem can be formulated as a MSGC problem.

It may be noted that the MSC problem is a generalization of the SC problem and the MSGC problem is a generalization of the MSC problem. As SC is a well known NP-complete problem (Kleinberg and Tardos 2006), it can easily verified that both MSC and MSGC problems are NP-Complete. However, unlike the SC problem for which a *log n* factor approximation algorithm exists, and for the MSC problem for which a $(1 - 1/e)$ factor approximation algorithm exists, in the following, we show that $1/k$ factor approximation algorithm ($k > 1$) for the MSGC problem cannot exist unless $P = NP$.

Theorem 5. *Unless $P = NP$, there cannot be a polynomial time approximation*

Set Cover	Maximum Set Group Cover
1. $A_{SC} = \{a_1, a_2, a_3\}$	1. $A_{MSGC} = \{a_1^1, a_2^1, a_3^1, a_1^2, a_2^2, a_3^2, a_1^3, a_2^3, a_3^3\}$
2. $\mathcal{A}' = \{A_1, A_2\}$	2. $\mathcal{A}' = \{A_1^1, A_1^2, A_1^3, A_2^1, A_2^2, A_2^3\}$
3. $A_1 = \{a_1, a_2\}$	3. $A_1^1 = \{a_1^1, a_2^1\}$, $A_1^2 = \{a_1^2, a_2^2\}$, $A_1^3 = \{a_1^3, a_2^3\}$
4. $A_2 = \{a_2, a_3\}$	4. $A_2^1 = \{a_2^1, a_3^1\}$, $A_2^2 = \{a_2^2, a_3^2\}$, $A_2^3 = \{a_2^3, a_3^3\}$
	5. $G = \{G_1, G_2, G_3\}$, $G_1 = \{a_1^1, a_2^1, a_3^1\}$, $G_2 = \{a_1^2, a_2^2, a_3^2\}$, $G_3 = \{a_1^3, a_2^3, a_3^3\}$

Table 25. Example of creation of an instance of Maximum Set-Group Cover Problem from an instance of Set Cover Problem

algorithm for the MSGC problem with a performance factor that guarantees for every instance I of the MSGC problem $APP(I) \geq \lceil OPT(I)/k \rceil$, where $APP(I)$ and $OPT(I)$ represents the approximate and optimal solutions respectively for the MSGC problem instance I and k is a real number with $k > 1$.

Proof: We claim that if such an algorithm existed, the Set Cover problem, which is known to be NP-complete, could have been solved in polynomial time. Suppose if possible, such an algorithm APP_{MSGC} exists. From an instance of the SC problem, given as $A_{SC} = \{a_1, ..., a_n\}$ and $\mathcal{A}'_{\mathcal{SC}} = \{A'_1, ..., A'_m\}$ $(A'_i \subseteq A_{SC}, 1 \leq i \leq m)$, we create an instance of the MSGC problem, by making n copies of the instance of the SC problem. Thus,

$$A_{MSGC} = \{a_1^1, \ldots, a_n^1, a_1^2, \ldots, a_n^2, \ldots, a_1^n, \ldots, a_n^n\} \tag{6.1}$$

$$\mathcal{A}'_{\mathcal{MSGC}} = \{A_1'^1, ..., A_1'^n, A_2'^1, ..., A_2'^n, \ldots, A_m'^1, ..., A_n'^m\} \tag{6.2}$$

$(A_j'^i \subseteq A_{MSGC}, 1 \leq i \leq n, 1 \leq j \leq n)$, $A_j'^i = \{a_k^i | a_k \in A_j, \forall i, 1 \leq i \leq n, 1 \leq j \leq m)$, $\mathcal{G}_{\mathcal{MSGC}} = \{G_1^{MSGC}, ..., G_n^{MSGC}\}$, and $G_i^{MSGC} = \{a_1^i, \ldots, a_n^i\}, \forall i, 1 \leq i \leq n)$. $(G_i^{MSGC} \subseteq A_{MSGC}, 1 \leq i \leq n)$.

An example of construction of an instance of the MSGC problem from an instance of the SC problem is shown in Table 25. Given an instance of the SC problem, using the MSGC instance creation rules above, we can create the corresponding instance of the MSGC problem. If there is a polynomial time algorithm APP_{MSGC} with $APP(I) \geq \lceil OPT(I)/k \rceil$, performance guarantee, we can apply it to the instance of the MSGC problem created from the instance of the SC problem. The algorithm will either return zero, implying that no group can be completely covered, or a non-zero number, implying that at least one group can be completely covered. If the algorithm returns zero, we can conclude that the SC problem has no solution. If the algorithm returns a non-zero number, it implies that the SC problem has a solution. Thus, we can conclude that if there exists a polynomial time approximation algorithm for the MSGC problem with a performance guarantee of $APP(I) \geq \lceil OPT(I)/k \rceil$, for some real number k, $k \geq 1$, then the SC problem, which is known to be NP-complete, can be solved in polynomial time. This implies that unless $P = NP$, no such polynomial time approximation algorithm can exist for the MSGC problem.

6.3.2 Optimal Solution for the BCICS Problem with ILP

The goal of BCICS is to have unique signature for as many nodes as possible, subject to the constraint that the number of nodes where sensor is placed does not exceed the specified budget, B.

Instance: A graph $G = (V, E)$ and an integer B.
Problem: Find a subset $V' \subseteq V$ of cardinality B (i.e., $|V'| = B$ such that placement of sensors at these nodes ensures that a largest subset of nodes V'' of V has a unique signature associated with it.

For each $v_i \in V$, we use an indicator variable x_i, such that

$$x_i = \begin{cases} 1, & \text{if a sensor is placed at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Also, for each $v_i \in V$, we use an indicator variable y_i, such that

$$y_j = \begin{cases} 1, & \text{if a } v_j \text{ ends up having a unique signature,} \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Maximize $\sum_{v_j \in V} y_j$

Budget Constraint: $\sum_{v_i \in V} x_i \leq B$,

In addition to the Budget Constraint, we introduce two additional constraints, *Coverage Constraint* and Unique Coverage Constraint. Before we introduce the constraints, we first define the terms *Coverage* and *Unique Coverage.*

Definition 6.3.8. Coverage of a node v_i is the *Closed Neighborhood Set* of node v_i, and is denoted by $Cov(v_i) = \{v_i \cup N^+(v_i)\}$.

Definition 6.3.9. Unique Coverage of a node pair (v_i, v_j) is the *Exclusive-OR* of the Closed Neighborhood Set of the nodes v_i and v_j and is denoted by $Uni\_Cov(v_i, v_j) = N^+(v_j) \bigoplus N^+(v_k)$

The Coverage and the Unique Coverage constraints are:

Coverage Constraint: $\forall v_j \in V$

$M_1 \times (1 - y_j) + \sum_{v_i \in N^+(v_j)} x_i \geq 1$,

Unique Coverage Constraint: $\forall v_j, v_k \in V, v_j \neq v_k$

$M_2 \times (2 - y_j - y_k) + \sum_{v_i \in \{N^+(v_j) \bigoplus N^+(v_k)\}} x_i \geq 1$

Note that the objective function ensures that the largest number of nodes in V receives a unique signature. The budget constraint ensures that not more than B nodes in V

can be selected for sensor placement. The Coverage Constraint ensures that if node v_i has a unique signature (i.e., $y_i = 1$), a sensor must be placed in at least one node in its closed neighborhood (as otherwise v_i will not have any signature, let alone a unique signature). The Unique Coverage Constraint ensures that for every pair of nodes (v_j, v_k) in V to have unique signatures associated with them, (i.e., $y_j = 1$ and $y_k = 1$), a sensor must have been placed in at least one node in the node set $N^+(v_j) \bigoplus N^+(v_k)$. This guarantees that v_j and v_k will not have identical signatures. The parameters M_1 and M_2 in the constraints are two large constants.

Dataset	Num Nodes	Num Edges	MICS Solution	$k = 25\%$		$k = 50\%$		$k = 75\%$	
				OPT	Cov. %	OPT	Cov. %	OPT	Cov. %
Fourteen Pipes	12	26	8	3	25%	7	58.33%	10	83.33%
Modified 19 Pipe	14	35	9	6	42.85%	9	64.28%	12	85.71%
Hanoi	32	66	21	10	31.25%	18	56.25%	25	78.12%
FOWM	45	94	30	14	31.11%	25	55.55%	37	82.22%
Kentucky 3	275	646	161	83	30.18%	160	58.18%	227	82.54%
Calibration	396	840	257	119	30.05%	228	57.57%	327	82.57%
Long Term	407	866	263	121	29.73%	234	57.49%	336	82.55%
Kentucky 2	812	1927	485	250	30.78%	476	58.62%	672	82.75%
Kentucky 1	859	1844	548	257	29.91%	496	57.74%	710	82.65%
Kentucky 4	962	2103	619	294	30.56%	561	58.31%	795	82.64%
Kentucky 8	1329	2936	826	412	31%	786	59.14%	1113	83.74%
Kentucky 12	2355	4810	1583	686	29.12%	1296	55.03%	1890	80.25%

Table 26. Optimal BCICS Results for Water Distribution Network Systems

6.4 Experimental Results

The datasets used in our experimentation were obtained from the Kentucky Water Resources Research Institute (Kentucky 2001). Each dataset contains information regarding the collection of pipes, pumps, valves, junctions, tanks, and reservoirs that make up a water distribution system. For our study, the points of interest are the nodes which represent junctions, tanks, and reservoirs whereas the edges represent pipes, pumps, and valves.

The results of our experimentation on the datasets are presented in Table 26. For the ease of understanding, we will describe in detail the result for the first row, i.e., the Fourteen Pipes dataset. This dataset has 12 nodes and 26 edges. The third column denotes the MICS cardinality for this graph. The rationale for computing the MICS for the datasets was to ensure that we varied the budget parameter according to the MICS cardinality. In our setup, we have considered three different budget parameters, $k = 25\%, 50\%$ and 75% of the MICS cardinality. For instance, in the Fourteen Pipes dataset, the MICS solution is 8 and the budget parameters are $k = 25\%$ of $8 = 2$, $k = 50\%$ of $8 = 4$ and $k = 75\%$ of $8 = 6$. Now, the task is to identify the locations for sensor placement which maximizes the number of nodes which would be uniquely covered (i.e., will have a unique fault signature). For the Fourteen Pipes dataset, with at most 2 sensors ($k = 25\%$), we see that the optimal number of nodes uniquely covered is 3, for which the coverage % (% of nodes uniquely covered with the budget) is $\frac{3}{12} = 25\%$, where 12 is the number of nodes in the network. Similarly, for $k = 50\%$ or when the budget is 4, we can uniquely cover 7 nodes optimally, with 58.33% coverage, and finally for $k = 75\%$, we can cover 10 nodes optimally, with 83.33% coverage. Similar results for the other datasets follow. Note that, for all the datasets considered, the coverage $\% \geq k$. In other words, the benefits, in terms of the number of nodes uniquely covered, outweighs the cost (budget). The average benefits for the costs $k = 25\%, 50\%, 75\%$ are 30.96%, 58.04% and 82.42%.

Figure 23 illustrates the time taken by the optimal approach. Note that the optimal solution computes the solution for the largest graph considered fairly quickly, i.e., in a couple of minutes.

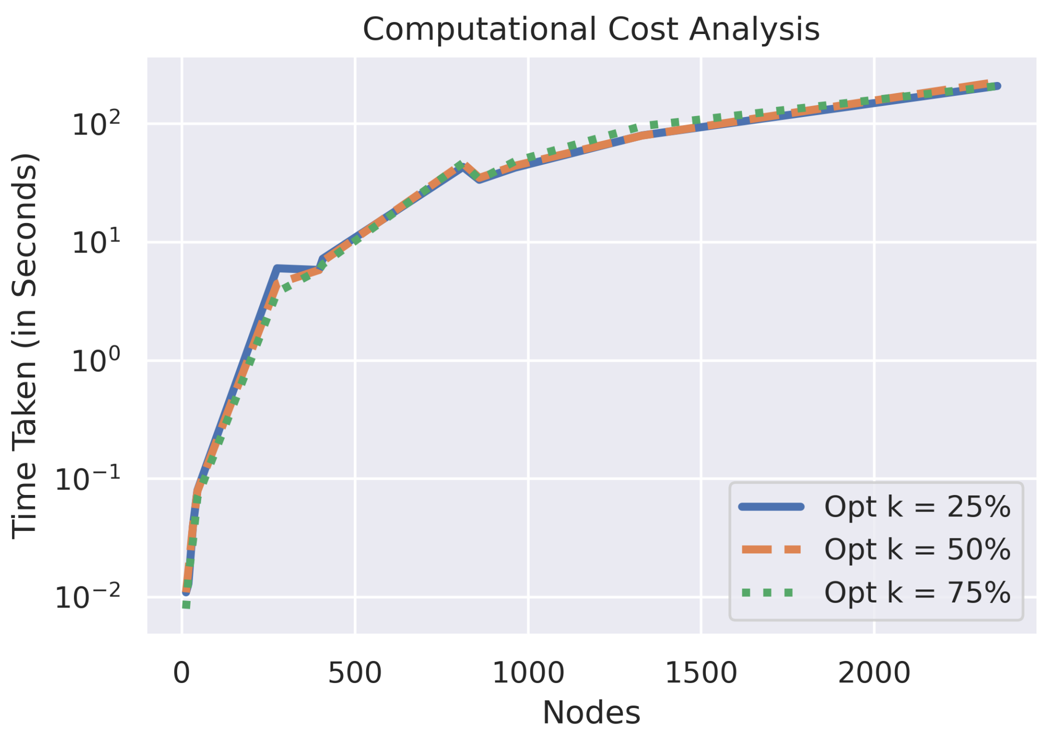


Figure 23. BCICS Computational Run Time

6.5 Conclusion and Future Directions

In this chapter we introduced a novel budget constrained version of the Identifying Code problem, geared towards identifying sources of anomalies in water distribution systems of large metropolitan cities. We provided an optimal solution for the problem through ILP and proved that no approximate algorithm for the MSGC with $1/k$ factor bound ($k \geq 1$) can exist, unless $P = NP$. Conventionally, ILPs tend to be computationally expensive, however, in our experimentation, computation times are fairly small, even for graphs with more than 2300 nodes and 4800 edges. It took less than a couple of minutes using GUROBI on an Intel i-9 processor with 128GB RAM.

Some of the future directions that could be examined in this domain are as follows:

- *Game Theoretic Monitoring Strategies*: As mentioned previously in the Human-Human Interaction Networks chapter, the problem discussed in this chapter could also be set up as a two-player game. This would dictate the movement of the sensors in the network so that the attacker has difficulties in figuring out the actual deployment locations and consequently failing in its attack.
- *Robust BCICS Problem*: Just as we discussed on a future robust Identifying Code problem, there also is an equal need on a future robust BCICS problem. The setting considered in this paper will fail to work if - (i) the sensor fails (due to an attack or mechanical failure), or (ii) the transmitted data from the sensor does not reach the control center.

Chapter 7

IDENTIFYING CODE PROBLEMS FOR RESTRICTED GRAPH CLASSES

7.1 Structural Health Monitoring

The Structural Health Monitoring (SHM) problem for critical infrastructures, such as bridges, buildings, electric power equipments, using Wireless Sensor Networks (WSNs), has received considerable attention in the research community in recent years (Noel et al. 2017). Sensors placed in the deployment area have two functions: (i) to sense a target function such as temperature, pressure, vibration, etc., and (ii) to transmit the sensed data either directly or through multiple other sensor nodes (which serves as relays) to the control station, for the analysis of the sensed data. While the first function relates to coverage of the sensing region, the second function relates to the connectivity aspects of the network formed by the sensors. The thrust of this chapter is on the coverage aspects of the sensors distributed over a geographic region. The sensors used for monitoring critical infrastructures may be *accelerometers, strain sensors, corrosion sensors, linear voltage differential transducers* and *optical fiber transducers* (Noel et al. 2017). Each of these sensors have a specific *sensing range* associated with it. For example, seismic sensors are deployed to detect earthquakes in regions of interest. All these sensors have certain sensitivity in the sense that some are more sensitive than others, for instance, acceleration sensors may be able to detect magnitude 1 earthquakes, within a few kilometers, whereas, highly sensitive seismometers will be able to detect earthquakes in a much larger distance. Hence, each of these sensors have a sensing range associated with it. A sensor can only

Figure 24. Structural Monitoring of Bridges

sense any abnormality if it happens within its sensing range, i.e., each sensor has a *coverage area* associated it. The coverage aspect of sensor networks alone has been studied extensively (Wang 2011; Cardei and Wu 2004). The survey paper on Coverage Problems in Sensor Networks (Wang 2011), references close to two hundred papers. Accordingly, a multitude of sensor coverage models, such as (i) Boolean Sector Coverage Model, (ii) Boolean Disc Coverage Model, (iii) Attenuated Disc Coverage Model, (iv) Truncated Attenuated Disc Models, (v) Estimation Coverage Models, etc. have been studied by various research groups (Wang 2011).

Cardei and Wu in (Cardei and Wu 2004), classified coverage problems into three broad classes, (i) Point Coverage, (ii) Area Coverage, and (iii) Barrier Coverage. While in the area coverage problem, an entire area (in two or three dimensional space) has to be sensed (monitored), in the point coverage problem, only a *specified set of points* (points of interest) in two or three dimensional space, has to be monitored. Moreover, oftentimes there are restrictions on the locations (in two or three dimensional space), where the sensors can be deployed. These are the potential locations where the sensors can be deployed, and these locations can be viewed as another set of specified points (in a two or three dimensional space). For example, accelerometers cannot be deployed in any arbitrary location, due to various constraints such as, cost of deployment, access difficulty, property rights, etc.

One of the most frequently studied problems in this context is the Sensor Placement Optimization (Kleinberg and Tardos 2006), whose goal is to find the smallest set of locations (among the potential locations for sensor deployment), so that all the points of interest can be sensed. In other words, every point of interest should be under the *coverage* area of at least one deployed sensor. If a boolean disc coverage model (Wang 2011) is used for sensor coverage, the Sensor Placement Optimization problem can be formulated as a Set Cover problem (Kleinberg and Tardos 2006) and a number of studies using this model are available in the literature (Kleinberg and Tardos 2006).

Although a number of studies on sensor placement optimization problem follow the set cover formulation to find a solution, it has a serious limitation on the *accurate* identification of the location, where some abnormality is detected by one or more of the deployed sensors. We illustrate this point with the help of an example. In Figure 25, the ten red points (numbered from 1-10) indicate the points to be sensed (monitored), the eight blue points (numbered from 11-18) indicate the potential locations where the sensors can be deployed and the green circles (centered on each blue point) indicate the coverage area of a sensor deployed at that blue point. In this example, it can

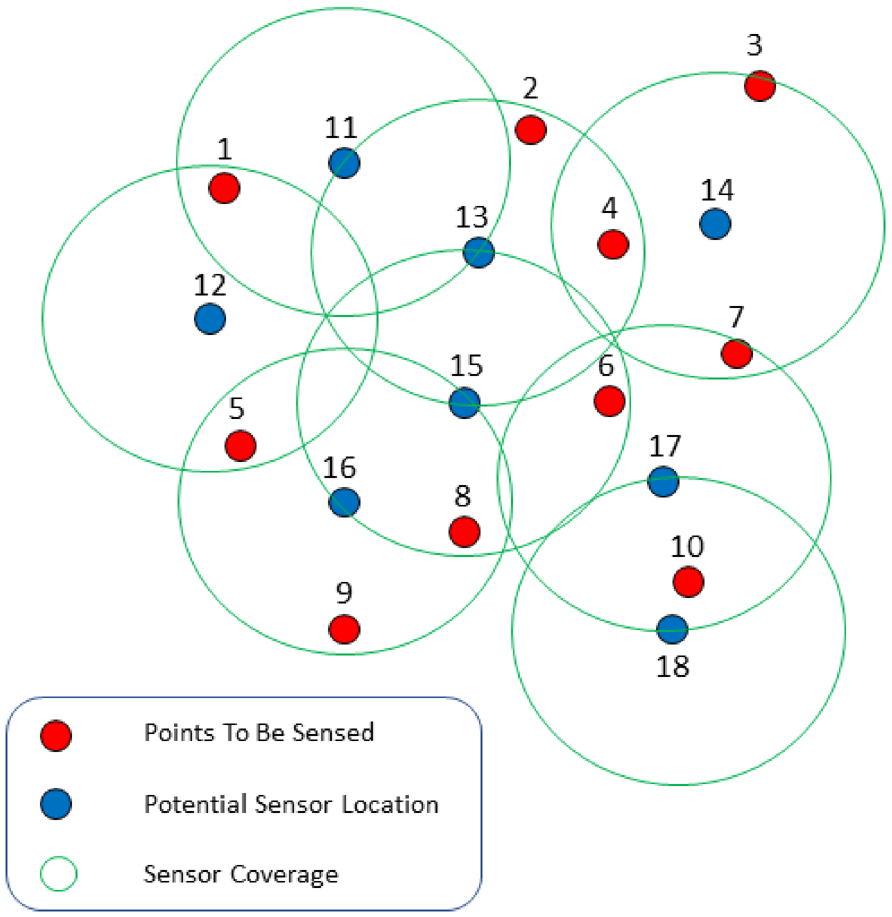


Figure 25. Potential Sensors and Sensing Locations

be verified that if sensors are deployed in locations 11, 13, 14, 16 and 17, all points from 1 to 10 will be within the sensing range of at least one sensor. Specifically, the points (1-10) covered (sensed) by the sensors $11, 13, 14, 16, 17$ are shown in Table 27. In Table 28, we present the sensors that are actually sensing the points 1-10, using the Set Cover approach.

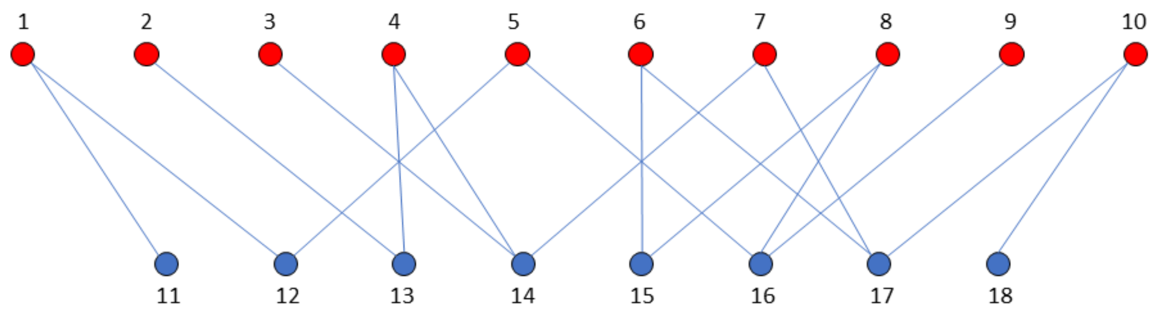


Figure 26. Bipartite Graph Corresponding to Potential Sensors and Sensing Locations

Sensor Location	Points Sensed	Sensor Location	Points Sensed
11	1	15	6, 8
12	1, 5	16	5, 8, 9
13	2, 4	17	6, 7, 10
14	3, 4, 7	18	10

Table 27. Points Covered by Each Sensor

Points Sensed	Sensor Location	Points Sensed	Sensor Location
1	11	6	17*
2	13	7	14, 17
3	14	8	16**
4	13, 14	9	16**
5	16**	10	17*

Table 28. Sensors Covering Each Point Following Set Cover

The serious limitation of the set cover based approach to optimal sensor placement problem is that, it may fail to uniquely identify the point where an abnormality is detected by the sensor. We elaborate this point with the results shown in Table 27 and Table 28. In this example, sensors were deployed at locations $11, 13, 14, 16, 17$ and this deployment ensured that all points to be sensed were within the coverage area of at least one sensor. Suppose the control center has five indicator lamps A, B, C, D, E corresponding to five sensors located at $11, 13, 14, 16, 17$. If the sensor does not sense

Points Sensed	Sensor Location	Points Sensed	Sensor Location
1	12	6	15, 17
2	13	7	14,17
3	14	8	15, 16
4	13, 14	9	16
5	12, 16	10	17

Table 29. Sensors Covering Each Point Following Identifying Code

an abnormality at the point it is sensing, then the corresponding lamp is lit green. If a sensor senses an abnormality at a point, then the corresponding lamp turns red. From Table 28, it can be seen that points 6 and 10 are sensed by sensor 17 only, and points 5, 8 and 9 are sensed by 16 only. The implication of this is that if lamp E (corresponding to sensor 17) turns red, then it will not be possible to ascertain if the abnormality was detected at point 6 or 10. Similarly, if lamp D (corresponding to sensor 16) turns red, then it will not be possible to ascertain if the abnormality was detected at point 5 or 8 or 9.

This limitation of failure to uniquely identify the point where abnormality is detected by the sensor, can be overcome by deployment of additional sensors. In this example, instead of deploying sensors at locations $11, 13, 14, 16, 17$, if they were deployed at locations $12, 13, 14, 15, 16, 17$, then each point would have been sensed in the way as shown in Table 29. It may be noticed that deployment of six sensors, instead of five, avoids the problem of failure of unique identification of points where abnormality is detected.

The mathematical foundation of computing the least number of sensors needed for unique identification of points where abnormality is detected, is provided by the concept of Identifying and Discriminating Codes (Karpovsky, Chakrabarty, and Levitin 1998; Charbit et al. 2006). Since its introduction, these codes have been studied fairly extensively for various classes of graphs. We show that the problem of computing the least number of sensors needed for unique identification of points, where abnormality is detected, for points in one-dimensional and two-dimensional space is equivalent to computation of *Minimum Discriminating Code Set (MDCS)* of *Unit Interval Bigraph* and *Unit Disc Bigraph* respectively. A major contribution of this chapter is the development of a polynomial time algorithm for the MDCS problem for Unit Interval Bigraphs. It may be noted that when the intervals associated with the nodes of the graph are of arbitrary size, then the MDCS problem is NP-Complete (Foucaud and Perarnau 2011). Computational complexity of related problems, such as Identifying Code for Interval Graphs is NP-Complete, whereas it is still open for Interval Graphs with unit length intervals (Bousquet et al. 2015).

7.1.1 Related Work

As noted earlier, sensor placement problems for monitoring critical infrastructures have been studied extensively in the last few years (Li, Wang, Ni, et al. 2009; Bhuiyan et al. 2014; Basu, Padhee, et al. 2018; Kim et al. 2007; Lynch and Loh 2006). A particular example of health monitoring in civil infrastructures using wireless sensor networks can be seen in (Kim et al. 2007), where the authors deployed sensors to monitor the Golden Gate Bridge (Figure 27). In this application, both the points to be monitored and the locations where the sensors can be deployed, lie on a single line, as shown in Figure 27. Our study on accurate identification of the location where abnormality is sensed, in which sensors and points of interest are located on one line, is motivated by the study in (Kim et al. 2007). WSN based SHM system is also deployed at the many bridges in China including the ZhengDian bridge (Noel et al. 2017). Aside from bridges, SHM systems have also been deployed for monitoring sports stadium, buildings and wind turbines (Noel et al. 2017).

Karpovsky *et al.* introduced the concept of Identifying Codes in (Karpovsky, Chakrabarty, and Levitin 1998) and provided results for Identifying Codes for graphs with specific topologies, such as binary cubes and trees. Using Identifying Codes, Laifenfeld *et al.* studied joint monitoring and routing in wireless sensor networks, in (Laifenfeld et al. 2009). Ray *et al.* studied location detection problem in emergency sensor networks, using Identifying Codes (Ray et al. 2004). In this work, they also introduced the concept of robust Identifying Codes to deal with faults in sensor networks. They presented an algorithm for generating *irreducible* Identifying Codes in polynomial time. It may be noted that irreducible Identifying Code is only a *minimal* Identifying Code and may not be the *minimum* (or optimal) Identifying Code. In contrast, we present an algorithm for construction of optimal Identifying Code for the problem scenario, where the points of interest and potential locations for sensor deployment, lies on a single line. Sen *et al.* studied monitoring terrorist networks using Identifying Codes (Arunabha Sen et al. 2018).

Algorithms and complexity of computation of Identifying Codes for restricted class of graphs such as, Interval and Permutation graphs, were studied in (Foucaud and Perarnau 2011; Foucaud 2015). An approximation algorithm for the computation of minimum Identifying Code for Interval graphs with a performance bound of six, was presented in (Bousquet et al. 2015). A special case, where only a subset of nodes needs a unique code, can be modeled with a bipartite graph, and this version of Identifying Codes is called “Discriminating Codes” and was studied in (Charbit et al. 2006, 2006). This special case is relevant for our study as, our problem formulation requires us to find the unique signatures of all nodes of one side of bi-partition of a bipartite graph, by selecting only a subset of the nodes in the other side of bi-partition. This formulation corresponds directly to “Discriminating Codes”. However, the bipartite graphs that appear in our formulation is not just any bipartite graph, but subsets of bipartite graphs known as Interval Bigraphs and Unit Disc Bigraphs, corresponding to our study of points being in one or two dimensional space respectively. (Müller and Sereni 2009) presented a polynomial time algorithm for recognition of Interval Bigraphs.

7.1.2 Problem Formulation

In Figure 25, the ten red points (numbered from 1-10) indicate the points to be sensed (monitored), the eight blue points (numbered from 11-18) indicate the potential locations where the sensors can be deployed and the green circles (centered on each blue point) indicate the coverage area of a sensor deployed at that blue point. From these set of red and blue points, we construct a graph using the following construction rules: (i) Corresponding to each red point, we have a red node, (ii) Corresponding to each blue point, we have a blue node and (iii) There is an edge between a blue node and a red node, if and only if the corresponding red point is within the green circle, centered at the corresponding blue point. The graph constructed from the problem instance in Figure 25, is shown in Figure 26. Clearly, the graph constructed by following the above construction rules will result in a Bipartite graph. However, a pertinent question in this regard is whether any Bipartite graph can be constructed by following the rules or the constructed graphs constitute only a *subset* of all Bipartite graphs. In the following theorem, we prove that the constructed graphs constitute only a *subset* of all Bipartite graphs.

In the following, we define a set of graphs that are relevant for this study.

Definition 7.1.1. Interval Bigraphs: An interval bigraph is an undirected bipartite graph $G = (V_1 \cup V_2, E)$, whose edge set is the intersection of the edge set of an interval graph with vertex set $V_1 \cup V_2$, and the edge set of a complete bipartite graph with bi-partition $V_1 \cup V_2$. A bipartite interval representation of an interval bigraph is given by a bipartitioned set of intervals for its vertices, such that vertices are adjacent if and only if the corresponding intervals intersect, and belong to opposite sides of the bi-partition (Müller and Sereni 2009).

Definition 7.1.2. Unit Disc Bigraphs: A unit disc bigraph is an undirected bipartite graph, whose edge set is the intersection of the edge sets of a unit disc graph and the edge set of a complete bipartite graph on the same vertex set. A bipartite unit disc representation of a unit disc bigraph is given by a bipartitioned set of circles for its vertices, such that vertices are adjacent if and only if the corresponding circles intersect and belong to opposite sides of the bi-partition.

Lemma 6. *The number of distinct regions created by intersection of n circles on a plane is at most $n^2 - n + 2$.*

Proof. Proof by induction. If $n = 1$ there are two regions (one inside and the other outside the circle), and the inductive hypothesis holds as $n^2 - n + 2 = 1 - 1 + 2 = 2$. Assume that the hypothesis is true as long as $n \leq m$. If a new $(m + 1)$-th circle is introduced, it can cross the old ones in at most 2m points. Each segment can cut an existing region in two parts, adding 2m regions. Thus the maximum number of regions after introduction of the for $(m + 1)$-th circle is $m^2 - m + 2 + 2m = (m + 1)^2 - (m + 1) + 2$. □

Theorem 7. *The graphs constructed by following the above construction rules only create a subset of all Bipartite graphs.*

Proof. The theorem can be proven by showing that there exists at least one Bipartite graph that cannot be constructed by following the construction rules.

Consider a Bipartite graph $G = (V_1 \cup V_2, E)$, where $V_1 \cup V_2$ constitutes the bipartition and cardinalities of V_1 and V_2 are n and $2^n - 1$ respectively. Each node in V_2 corresponds to a *non-empty* subset of the nodes in V_1 and is connected by an edge $e \in E$ to those nodes.

Claim: Such a Bipartite graph cannot be constructed using the graph construction rules given above. The reason such a graph cannot be constructed is the following. The node sets V_1 and V_2 corresponding to two sets of points on a two dimensional plane. Suppose that we refer to the set of points corresponding to the nodes in V_1 as red points and the set of points corresponding to the nodes in V_2 as blue points. Suppose that we draw a unit radius circle with each red point as the center. Since each node $v \in V_2$ corresponds to a distinct non-empty subset of nodes of V_1 and is connected to this subset of nodes, in order to satisfy the graph construction rules, the blue point corresponding to $v \in V_2$ must be located in a distinct region created by intersection of n circles corresponding to n red points. However, from Lemma 6 we know that the number of distinct regions created by intersection of n circles on a plane is at most $n^2 - n + 2$. For existence of the graph $G = (V_1 \cup V_2, E)$, $2^n - 1$ distinct regions are needed as the locations of the $2^n - 1$ blue points. As $2^n - 1 > n^2 - n + 2$ for $n \geq 4$, the graph $G = (V_1 \cup V_2, E)$, such locations cannot be found when $n \geq 4$ and hence the graph cannot be constructed using the graph construction rules given above. □

Theorem 8. *The graph constructed from the two sets of points (red and blue) on a* *two dimensional plane using the construction rules given above will be an Unit Disc Bigraph.*

Proof. Clearly, the graph constructed following the construction rules will be a bipartite graph $G = (V_1 \cup V_2, E)$, where V_1 is the set of nodes corresponding to the set of red points and V_2 is the set of nodes corresponding to the set of blue points. Suppose that G_{udg} is a unit disc graph corresponding to the set of red and blue points, and G_{cbp} is a complete bipartite graph with bi-partition V_1 and V_2. It can be easily verified that $G = (V_1 \cup V_2, E)$ is the *intersection* of G_{udg} and G_{cbp}. If $G_{udg} = (V_1 \cup V_2, E_{udg})$ and $G_{cbp} = (V_1 \cup V_2, E_{cbp})$, then $G = (V_1 \cup V_2, E_{udg} \cap E_{cbp})$. $\square$

Theorem 9. *The graph constructed from the two sets of points (red and blue) on an one dimensional plane using the construction rules given above will be an Interval Bigraph.*

Proof. As in Theorem 8, the graph constructed following the construction rules will be a bipartite graph $G = (V_1 \cup V_2, E)$. Suppose that G_{ig} is an interval graph corresponding to the set of red and blue points and G_{cbp} is a complete bipartite graph with bi-partition V_1 and V_2. It can be easily verified that $G = (V_1 \cup V_2, E)$ is the intersection G_{ig} and G_{cbp}. $\square$

In this chapter, we focus on two deployment scenarios. In the first case, both the points of interest and the potential sensor deployment locations are on a two dimensional space (i.e., a plane). In the second case, they are located on an one dimensional space (i.e., a line). The motivation for considering the scenario where they are located on an one dimensional space comes from the fact that, in a large number of structures (such as long spanning bridges), both the locations to be sensed and the locations where sensors can be deployed, lie on a line. In a study undertaken

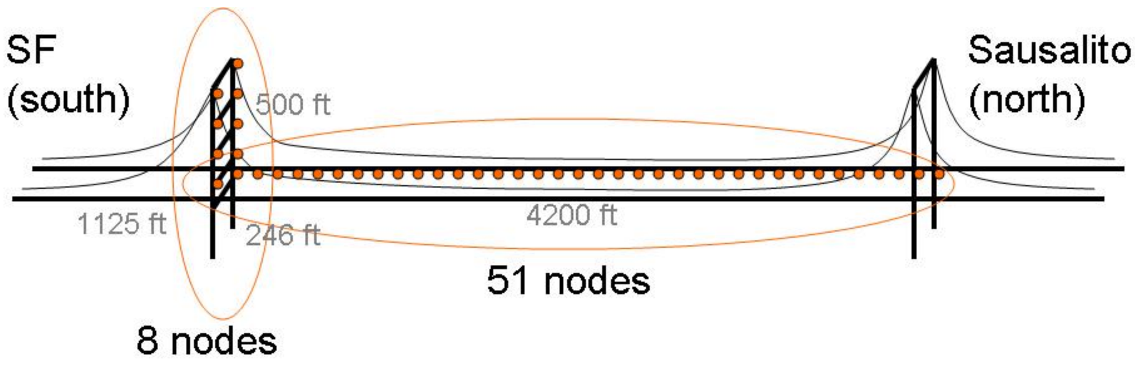


Figure 27. Golden Gate Wireless Sensor Network

by a University of California Berkeley team, a wireless sensor network (WSN) was deployed for structural health monitoring (SHM) of the Golden Gate bridge, where the sensors were placed on a line as shown in Figure 27. Similar efforts were undertaken in China as reported in (Noel et al. 2017).

It may be recalled that the main objective of this study is, to determine the *least* number of sensors and their locations, required for *unique identification* of the point of interest, where abnormality is detected. It may be noted that, in this study we focus our attention only to the scenario where abnormality (or failure) is restricted to only one point. In this chapter, we do not consider multiple simultaneous failure.

It is clear from the discussion on the graph construction rules for the points deployed in two or one dimensional spaces that, the corresponding graphs will be a *Unit Disc Bigraph* and a *Interval Bigraph* respectively. It may be recalled that, the MDCS Problem finds the smallest subset $V_2' \subseteq V_2$, such that injection of colors at these nodes, ensures that each node $v \in V_1$, receives a unique color through seepage. Thus, the objective of determining the *least* number of sensors required for *unique identification* of the point of failure can be realized, by the computation of MDCS for the Unit Disc Bigraph and Interval Bigraph, respectively.

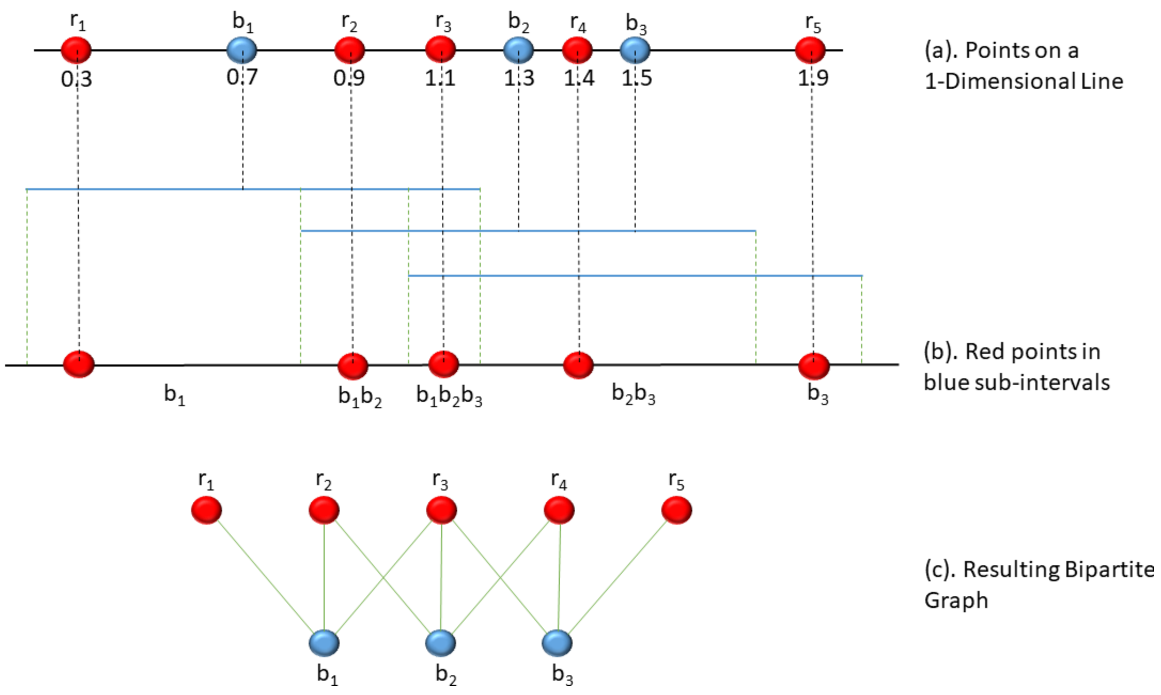

r_1
b_1
r_2
r_3
b_2
r_4
b_3
r_5
0.3
0.7
0.9
1.1
1.3
1.4
1.5
1.9
(a). Points on a 1-Dimensional Line
b_1
b_1b_2
$b_1b_2b_3$
b_2b_3
b_3
(b). Red points in blue sub-intervals
(c). Resulting Bipartite Graph

Figure 28. Bipartite Graph Generated From One Dimensional Points

7.1.3 Problem Solution

In this section, we first present a dynamic programming based optimal algorithm, for the computation of MDCS for points located in one dimensional space. We then present an Integer Linear Program for the computation of the MDCS for points located in two dimensional space.

7.1.3.1 One Dimensional Case

In this subsection, we provide a dynamic programming based algorithm to find the optimal solution for the problem in one dimension. Given a set of red points $R = \{r_1, ...r_n\}$ and a set of blue points $B = \{b_1, ..., b_m\}$ on a line, as shown in Figure 28, the dynamic programming algorithm finds the smallest subset $B' \subset B$, such that, injection of colors at these points, assigns a *unique* color (atomic or composite) to every red point in R through seepage. It may be recalled that the set of red points correspond to the points of interest and the set of blue points are potential locations for placement of sensors. The sensing range of a sensor is 1 unit, which implies that a blue point $b_i \in B$ can sense all the red points within the interval $b_i - 1$ to $b_i + 1$. The graph constructed from the set of points in Figure 28, following the graph construction rules mentioned earlier, is shown in Figure 28.

We use the notation R_j to be the $j-$th prefix of R, i.e., $R_j = \{r_1, ...r_j\}$. Similarly, B_i to be the $i-$th prefix of B, i.e., $B_i = \{b_1, ...b_i\}$. Since, the objective of this problem is to assign a *unique* color to all elements of R by injecting colors at the fewest number of elements of B, we refer to the R points as the *problem space* and B points as the *solution space.* We use the notation $D_{i,j}$ to indicate the smallest (optimal) subset of B_i that uniquely color all the elements of R_j. *In case there are multiple subsets $B'_i \subset B_i$ that uniquely color all the elements of R_j, the subsets B'_i are lexicographically ordered and for $D_{i,j}$, we store the subset B'_i that appears last in the lexicographical ordering.* We build a $m \times n$ table row by row and at the end of the completion of the table, the (m, n)-th entry provides the solution to the problem.

If no solution exists for problem space R_j with solution space B_i, we denote it by setting $D_{i,j} = \emptyset$. Prior to computation of $D_{i,j}$, all elements of the table from rows 1 to $i - 1$ have been computed. Moreover, all elements in the i-th row from columns 1 to $j - 1$ have also been computed. Only $j + 1$ of these elements can be candidates for $D_{i,j}$.

- Case 1: $D_{i-1,j}$, as this provides the smallest subset of B_{i-1} that uniquely colors all elements of R_j.

- Case 2: $D_{i-1,j-1} \cup b_i$, if $D_{i-1,j-1} \neq \emptyset$ and b_i uniquely colors r_j. ($D_{i-1,j-1} \neq \emptyset$ provides a non empty subset of B_{i-1} that assigns unique colors to R_{j-1}. Since b_i uniquely colors r_j, $D_{i-1,j-1}$ together with b_i, uniquely colors R_j).
- Case 3: $D_{i,j'}$, where $D_{i,j'}$ uniquely colors R_j, and j' is the value of j'', such that $|D_{i,j'}|$ is the smallest among all $|D_{i,j''}|, 1 \leq j'' < j$. (It may be noted that $D_{i,j'}$ guarantees unique color assignment to $R_{j'}$ with the smallest subset B'_i of B_i. However, it is possible that for some structure of the graph, not only does B'_i assign a unique coloring to $R_{j'}$, but also it assigns a unique coloring to R_j, where $j' < j$. As such, this should be considered as a candidate for $D_{i,j}$.)

It may be noted that while cases 1 and 2 produce one candidate each, case 3 produces 1 candidate out of $j-1$ possible candidates, for $D_{i,j}$.

With discussions above, $D_{i,j}$ can be expressed in the form of the following recurrence relation,

$$D_{i,j} = min(Set_1, Set_2, Set_3) \tag{7.1}$$

where, Set_1, Set_2, Set_3 are defined as follows:

$$Set_1 = \ D_{i-1,j}$$

$$Set_2 = min \begin{cases} D_{i-1,j-1} \cup \{b_i\}, & if\ D_{i-1,j-1} \neq \emptyset \text{ and } b_i \text{ uniquely colors } \ r_j \\ \emptyset, & \text{otherwise} \end{cases}$$

$$Set_3 = min \begin{cases} min_{\forall j'<j} D_{i,j'}, & if\ D_{i,j'} \text{ assigns unique colors to } R_j \\ \emptyset, & \text{otherwise} \end{cases}$$

Table Entry	Comparison	Result	Color
D_{11}	$\min(D_{01}, D_{00} \cup \{b_1\}, D_{10}) = \min\ (\emptyset, \{b_1\}, \emptyset)$	$\{b_1\}$	[A, *, *, *, *]
D_{12}	$\min(D_{02}, D_{01} \cup \{b_1\}, D_{11}) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{13}	$\min(D_{03}, D_{02} \cup \{b_1\}, min(D_{11}, D_{12})) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{14}	$\min(D_{04}, D_{03} \cup \{b_1\}, min(D_{11}, D_{12}, D_{13})) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{15}	$\min(D_{05}, D_{04} \cup \{b_1\}, min(D_{11}, D_{12}, D_{13}, D_{14})) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{21}	$\min(D_{11}, D_{10} \cup \{b_2\}, D_{20}) = \min\ (\{b_1\}, \emptyset, \emptyset)$	$\{b_1\}$	[A, *, *, *, *]
D_{22}	$\min(D_{12}, D_{11} \cup \{b_2\}, D_{21}) = \min\ (\emptyset, \{b_1 b_2\}, \emptyset)$	$\{b_1 b_2\}$	[A, AB, *, *, *]
D_{23}	$\min(D_{13}, D_{12} \cup \{b_2\}, min(D_{21}, D_{22})) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{24}	$\min(D_{14}, D_{13} \cup \{b_2\}, min(D_{21}, D_{22}, D_{23})) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{25}	$\min(D_{15}, D_{14} \cup \{b_2\}, min(D_{21}, D_{22}, D_{23}, D_{24})) = \min\ (\emptyset, \emptyset, \emptyset)$	$\emptyset$	[*, *, *, *, *]
D_{31}	$\min(D_{21}, D_{20} \cup \{b_3\}, D_{30}) = \min\ (\{b_1\}, \emptyset, \emptyset)$	$\{b_1\}$	[A, *, *, *, *]
D_{32}	$\min(D_{22}, D_{21} \cup \{b_3\}, D_{31}) = \min\ (\{b_1 b_2\}, \emptyset, \emptyset)$	$\{b_1 b_2\}$	[A, AB, *, *, *]
D_{33}	$\min(D_{23}, D_{22} \cup \{b_3\}, min(D_{31}, D_{32})) = \min\ (\emptyset, \{b_1 b_2 b_3\}, \emptyset)$	$\{b_1 b_2 b_3\}$	[A, AB, ABC, BC C]
D_{34}	$\min(D_{24}, D_{23} \cup \{b_3\}, min(D_{31}, D_{32}, D_{33})) = \min\ (\emptyset, \emptyset, \{b_1 b_2 b_3\})$	$\{b_1 b_2 b_3\}$	[A, AB, ABC, BC, C]
D_{35}	$\min(D_{25}, D_{24} \cup \{b_3\}, min(D_{31}, D_{32}, D_{33}, D_{34})) = \min\ (\emptyset, \emptyset, \{b_1 b_2 b_3\})$	$\{b_1 b_2 b_3\}$	[A, AB, ABC, BC, C]

Table 30. Computation of the Entries of the Table

	r_1	$r_1 r_2$	$r_1 r_2 r_3$	$r_1 r_2 r_3 r_4$	$r_1 r_2 r_3 r_4 r_5$
b_1	$\{b_1\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$b_1 b_2$	$\{b_1\}$	$\{b_1 b_2\}$	$\emptyset$	$\emptyset$	$\emptyset$
$b_1 b_2 b_3$	$\{b_1\}$	$\{b_1 b_2\}$	$\{b_1 b_2 b_3\}$	$\{b_1 b_2 b_3\}$	$\{b_1 b_2 b_3\}$

Table 31. Table Entries For the Figure

In the following, we present an algorithm for MDCS computation for points in one dimensional space, which is based on the recurrence relation given in Equation 7.1. The algorithm builds a $m \times n$ table, where m and n represents the number of blue and red points respectively, by filling out entries of the table row by row and column by column.

Theorem 10. *Algorithm 4 finds the optimal solution for the MDCS problem for the points in one-dimensional space.*

Proof. $D_{i,j}$ is the smallest subset B'_i of B_i that uniquely colors R_j. In Algorithm 4, $D_{i,j}$ is computed as the minimum of three sets, Set_1, Set_2, Set_3. One important requirement for any candidate (i.e., a subset of B_i) to be considered as $D_{i,j}$ is that, this subset has to uniquely color R_j.

Algorithm 4 MDCS Computation Algorithm (One Dimension)

```
1: Input: A set of red $R = \{r_1, ..r_n\}$ and blue $B = \{b_1, ..b_m\}$ points on a line
2: Output: The smallest subset $B' \subseteq B$ that uniquely colors $R$ ($D_{m,n}$).
3: $\forall i, j, D_{i,0} = D_{0,j} = \emptyset$
4: for i = 1 to m do
5:     for j = 1 to n do
6:         $Set_1 = D[i-1, j]$
7:         if $D[i-1, j-1] \neq \emptyset$ and $b_i$ uniquely colors $r_j$ then
8:             $Set_2 = D[i-1, j-1] \cup b_i$
9:         else
10:            $Set_2 = \emptyset$
11:        end if
12:        count = 0, max = 1000000, idx = 0
13:        for k = 1 to j - 1 do
14:            if $D[i, k]$ uniquely colors $R_j$ then
15:                count = count + $|D[i, k]|$
16:                if $|D[i, k]| < max$ then
17:                    max = $|D[i, k]|$
18:                    idx = $k$
19:                end if
20:            end if
21:        end for
22:        if count == 0 then
23:            $Set_3 = \emptyset$
24:        else
25:            $Set_3 = D[i, idx]$
26:        end if
27:        $D_{i,j} = min(Set_1, Set_2, Set3)$
28:    end for
29: end for
```

1. The set $D_{i-1,j}$ uniquely colors R_j, using the smallest subset of B_{i-1}. Since B_{i-1} is a subset of B_i, $D_{i-1,j}$ is a potential candidate for $D_{i,j}$. This is captured in Set_1.
2. The set $D_{i-1,j-1}$ uniquely colors R_{j-1} using the smallest subset of B_{i-1}. However, $D_{i-1,j-1}$, in addition to uniquely coloring R_{j-1}, may or may not uniquely color r_j. If $D_{i-1,j-1}$, in addition to uniquely coloring R_{j-1}, uniquely colors r_j, then problem reduces to Set_1. However, if $D_{i-1,j-1}$, does not uniquely color r_j, but augmentation of the set $D_{i-1,j-1}$, with b_i, enables r_j to receive a unique color, then this augmented set becomes a potential candidate for $D_{i,j}$. This is captured in Set_2.
3. For all $j' < j$, the set $D_{i,j'}$ uniquely colors $R_{j'}$, using the smallest subset of B_i. However, $D_{i,j'}$, in addition to uniquely coloring $R_{j'}$, may or may not

uniquely color R_j. Among the set of non-empty $D_{i,j'}$ that uniquely colors R_j, $1 \leq j' \leq j-1$, the one that has the smallest cardinality is a candidate for $D_{i,j}$. This is Set_3.

$D_{i,j}$ is the smallest subset B'_i of B_i that uniquely colors R_j. 1, 2 and 3 correspond to the following scenarios, (i) $b_i \notin B'_i, B'_i \subset B_i$, (ii) $b_i \in B'_i, B'_i \subset B_i$, and (iii) b_i may or may not be $\in B'_i, B'_i \subset B_i$, depending on whether it satisfies the required condition. Accordingly, these three scenarios are exhaustive from which potential candidates for $D_{i,j}$ must emerge. Since the algorithm examines all three sets, it is guaranteed to find the optimal solution. □

Lemma 11. *MDCS problem for Unit Interval Bigraph generated from m blue points and n red points has no solution in case $n \geq 2m$.*

Proof. There will be m intervals corresponding to m blue points. These intervals may be overlapping with each other, creating a set of subintervals, as shown in Figure 28. With m blue points, there can be at most $2m-1$ subintervals. If more than one red point appears in one blue subinterval, then the corresponding red nodes will have exactly identical neighborhood of blue nodes. Accordingly, these red nodes will be "twins". As noted earlier, the necessary and sufficient condition for a graph to have an Identifying Code is that the graph should be "twin" free. If the number of red points n is $2m$ or higher, by Pigeon Hole Principle, then at least one of the blue subintervals must have more than one red point. Accordingly, the corresponding graph will have "twins" and MDCS for the graph will not exist. □

Theorem 12. *Complexity of Algorithm 4 is $\mathcal{O}(m^5)$.*

Proof. The Algorithm 4 builds up a $m \times n$ table. The amount of computation involved in filling out (i, j)-th entry of the table is $\mathcal{O}(j)$, as it has to compute the minimum of the entries $D[i,1],...,D[i,j-1]$ and check if the red nodes in R_j received a unique color. The computation involved in checking for uniqueness is $\mathcal{O}(n^2)$. Accordingly, the complexity of the algorithm is $\mathcal{O}(mnjn^2)$. Since $j \leq n$ and $n \leq 2m-1$, the complexity of the algorithm is $\mathcal{O}(m^5)$, where m is the number of blue points, i.e., the potential sensor locations. $\square$

Example: Figure 28 illustrates 5 red points and 3 blue points in one dimension along with the corresponding bipartite graph. The solution to this problem, using Algorithm 4, can be found in the entry $D_{3,5}$ of Table 31.

Computational process through which the entries of Table 31 was found is shown in Table 30. Algorithm 4 computes $D_{i,j}$ by finding minimum of the three sets Set_1, Set_2, Set_3. It may be noted that, in this example, all three sets (Set_1, Set_2, Set_3) were used for filling out the entries of $D_{i,j}$ for different values of i, j. Specifically, the entries $D_{2,1}, D_{3,1}, D_{3,2}$ were filled using Set_1, $D_{1,1}, D_{2,2}, D_{3,3}$ were filled using Set_2, and $D_{3,4}, D_{3,5}$ were filled using Set_3. The last column of the Table 30 shows the color assignments in the five red nodes $r_1, ..., r_5$. For example, corresponding to the entry $D_{1,1}$, when the node b_1 is injected with a color, only the node r_1 receives the color through seepage from b_1. Similarly, corresponding to the entry $D_{3,2}$, when the nodes b_1, b_2 are injected with two distinct colors (denoted by the alphabets A, B), the node r_1 is colored A, whereas r_2 receives AB. The symbol $*$ indicates that the corresponding r point either did not receive a color or was outside the problem space at that stage.

7.1.3.2 Two Dimensional Case

In this subsection, we provide an Integer Linear Program (ILP) to find the optimal solution for the problem in two dimensions. Given a set of red points $R = \{r_1, r_2, ..., r_n\}$

and a set of blue points $B = \{b_1, b_2, ..., b_m\}$ on a two-dimensional plane, the Integer Linear Program illustrated below, finds the smallest subset $B' \subseteq B$. such that, injection of colors at these points, assigns a *unique* color to every red point in R, through seepage. It may be recalled that the construction of a graph from a set of points, has been described in subsection 7.1.2.

Instance: $G = (V_1 \cup V_2, E)$, an undirected bipartite graph.

Problem: Find the smallest subset $V_2' \subseteq V_2$, such that injection of colors at these nodes, ensures that each node $v_i \in V_1$, receives a unique color (either atomic or composite) through seepage.

We use the notation $N(v_i)$ to denote the neighborhood of v_i, for any $v_i \in V_1 \cup V_2$. Corresponding to each $v_i \in V_2$, we use an indicator variable x_i,

$$x_i = \begin{cases} 1, & \text{if a color is injected at node } v_i, \\ 0, & \text{otherwise} \end{cases}$$

Objective Function: Minimize $\sum_{v_i \in V_2} x_i$

Coloring Constraint: $\sum_{v_i \in N(v_j)} x_i \geq 1, \forall v_j \in V_1$

Unique Coloring Constraint:

$$\sum_{v_i \in \{N(v_j) \bigoplus N(v_k)\}} x_i \geq 1, \forall v_j, v_k \in V_1, v_j \neq v_k$$

$N(v_j) \bigoplus N(v_k)$ denotes the Exclusive-OR of the node sets $N(v_j)$ and $N(v_k)$. It may be noted that the objective function ensures that the fewest number of nodes in V_2 are assigned a color. The Coloring Constraint ensures that every node in V_1 receives at least one color through seepage from the colors injected at nodes in V_2.

A consequence of the Coloring Constraint is that, a node in V_1 may receive more than one color through seepage from the colors injected at nodes in V_2. The Unique Coloring Constraint ensures that, for every pair of nodes (v_j, v_k) in V_1, at least one node in the node set $N(v_j) \bigoplus N(v_k) \subseteq V_2$ is injected with a color. This guarantees that v_j and v_k will not receive identical colors through the color seepage from the nodes in V_2.

7.1.4 Experimental Results

In this subsection, we evaluate the performance of the ILP for the 2D MDCS problem. Conventionally, ILPs tend to be computationally expensive. However, we show that for the 2D MDCS problem, the computation times are fairly small, even for a graph with more than 64000 nodes and 155000 edges, it only took slightly more than 3 minutes. To simulate the performance, we generated random points on a 2D plane and constructed graphs following the rules illustrated in subsection 7.1.2. Table 32 illustrates the details of the various graphs and the time taken for computation of B'. The results were computed using the GUROBI Optimization package on an Intel i-5 with 32GB RAM.

7.2 Soccer Ball Graph Analysis

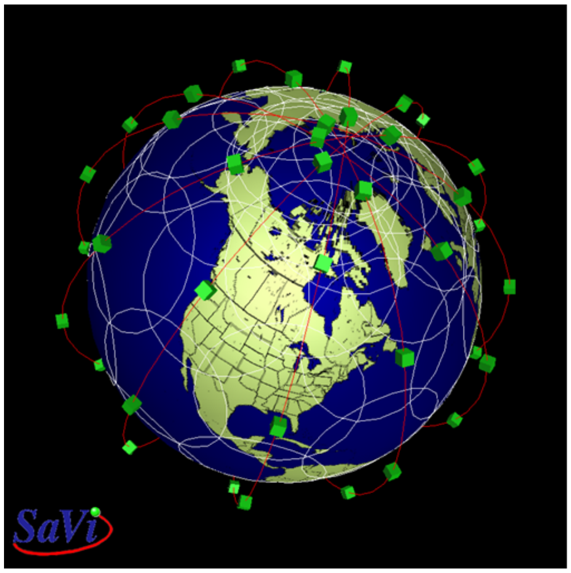
In this section, we study an event monitoring problem with satellites as sensors. The events that we focus on may be environmental (drought/famine), social/political (social unrest/war) or extreme events (earthquakes/tsunamis). Such events take place in *regions* on the surface of the earth, where a region may be a continent, a country,

$\lvert Nodes \rvert$	$\lvert Edges \rvert$	$\lvert R \rvert$	$\lvert B \rvert$	$\lvert B' \rvert$	$Time(s)$
45	105	5	40	3	0.06
89	187	7	82	4	0.60
170	276	14	156	10	1.59
422	1080	10	412	6	2.75
367	272	9	358	5	2.24
882	427	64	818	55	5.68
5927	3655	155	5772	106	25.18
64655	155339	8999	55656	6020	180.45

Table 32. 2D MDCS Computation Results

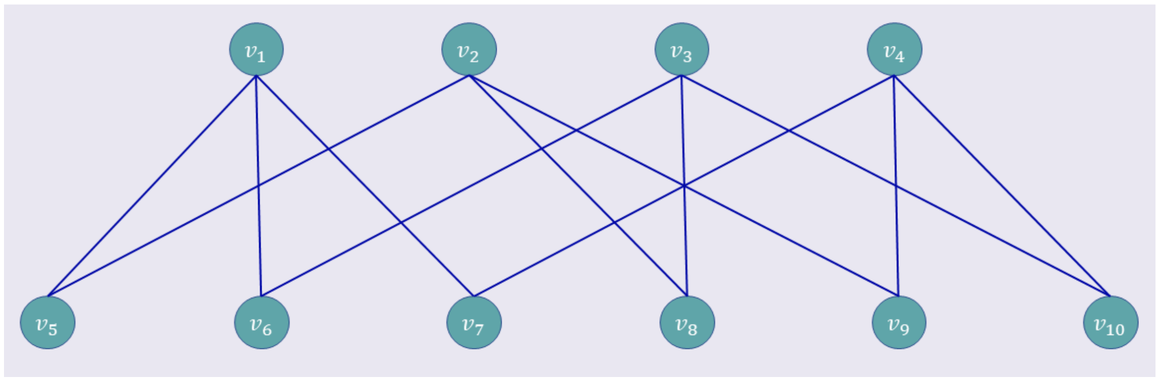
or a set of neighboring countries. The sensors that we envisage for monitoring such events are satellites placed in orbits surrounding the earth. A satellite constellation that can be deployed for such monitoring purposes is shown in Figure 29a. Examples of such constellations include the Global Positioning System (GPS) for navigation, the Iridium and Globalstar satellite telephony, and the Disaster Monitoring Constellation (DMC) for remote sensing. In particular, DMC is designed to provide earth imaging for disaster relief and was used extensively to monitor the impact of the Indian Ocean Tsunami in December 2004, Hurricane Katrina in August 2005, and several other floods, fires and disasters. The problem that we address in this paper is directly relevant to the services being provided by organizations such as the DMC.

For modeling the Earth's spherical structure, we use a soccer ball as a model of the planet Earth. In technical terms, a standard soccer ball is a *truncated icosahedron* with 12 pentagonal and 20 hexagonal patches (Kotschick 2006) (shown in Figure 29b as black and white patches). We associate a patch on the surface of the ball with a region on the surface of the Earth. Accordingly, in our model the surface of the Earth is partitioned into 32 regions. We assume that the coverage area of a satellite corresponds to a patch (region) and events are confined to a region. With such a



(a) Satellite Constellation Covering Earth

(b) Truncated Icosahedron

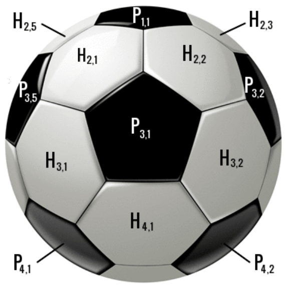


(c) Graph with Identifying Code Set $\{v_1, v_2, v_3, v_4\}$

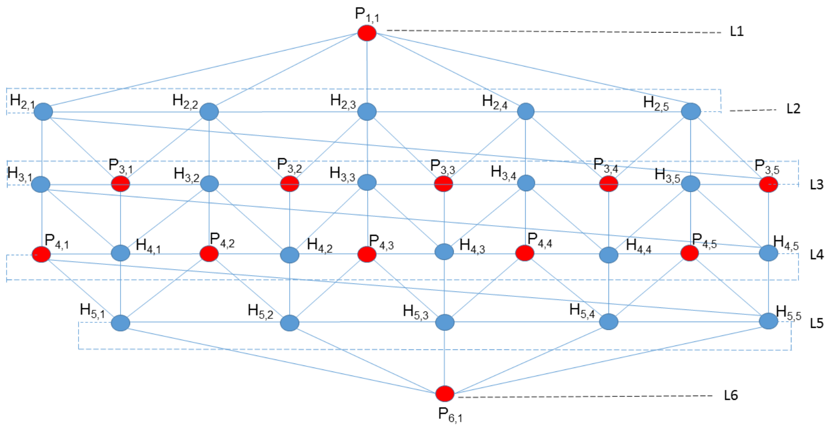
Figure 29. Satellites as Sensors and Soccer Ball as a Model of Planet Earth

framework, it is clear that with 32 satellites (one per region), all the 32 regions can be effectively monitored. However, if we assume that the *impact* of an event in one region will *spill* into the neighboring regions, and as such there will be indicators of such events in neighboring regions, then a significantly lower number of satellites may be sufficient for effective monitoring of all the regions. As an example of impact of an event spilling over to neighboring regions, one can think of a situation where war breaking out in one region can trigger an exodus of refugees to the neighboring regions. As these sensors are expensive, one would like to deploy as few sensors as possible, subject to the constraint that all the regions can be effectively monitored. In the following, we discuss *Identifying Codes* (Karpovsky, Chakrabarty, and Levitin 1998) that can be utilized for this purpose. In particular, we will show that 10 satellites are *sufficient* to *effectively monitor* 32 regions in the sense that, if an event breaks out in a region, that region can be *uniquely identified.* In fact there exists 26 different ways of deploying 10 satellites that will achieve the effective monitoring task. Moreover, we will establish that the effective monitoring task cannot be accomplished by deployment of *fewer than* nine satellites.

The notion of *Identifying Codes* (Karpovsky, Chakrabarty, and Levitin 1998) has been established as a useful concept for optimizing sensor deployment in multiple domains. From the soccer ball (Figure 30a), we construct a graph (referred to as a Soccer Ball Graph, SBG) where each of the 32 regions is represented as a node and two nodes have an edge between them if the corresponding regions share a boundary. The construction rules for the SBG are provided later and a two dimension layout of the SBG is shown in Figure 30b. We establish that the upper and lower bounds of the MICS problem for the SBG are 10 and nine respectively. Furthermore, we also es-



(a) Nodes Corresponding to the Patches on the Soccer Ball



$P_{1,1}$
L1
$H_{2,1}$ $H_{2,2}$ $H_{2,3}$ $H_{2,4}$ $H_{2,5}$
L2
$H_{3,1}$ $P_{3,1}$ $H_{3,2}$ $P_{3,2}$ $H_{3,3}$ $P_{3,3}$ $H_{3,4}$ $P_{3,4}$ $H_{3,5}$ $P_{3,5}$
L3
$P_{4,1}$ $H_{4,1}$ $P_{4,2}$ $H_{4,2}$ $P_{4,3}$ $H_{4,3}$ $P_{4,4}$ $H_{4,4}$ $P_{4,5}$ $H_{4,5}$
L4
$H_{5,1}$ $H_{5,2}$ $H_{5,3}$ $H_{5,4}$ $H_{5,5}$
L5
L6
$P_{6,1}$

(b) Soccer Ball Graph

Figure 30. Soccer Ball and the Corresponding Graph

tablish that there exist at least 26 different Identifying Code Sets of size 10 in the SBG.

7.2.1 Problem Formulation

A Soccer Ball Graph (SBG) $G = (V, E)$ is defined in the following way. The graph comprises of 32 nodes and 90 edges. The 32 nodes correspond to 32 patches (20 hexagonal and 12 pentagonal) of the soccer ball and two nodes in the graph have an edge between them if the corresponding patches share a boundary. A graph can have different layouts on a two dimensional plane. We show one layout of the SBG in Figure 30b where the nodes are labeled using a set of rules. The soccer ball, placed on a two dimensional plane (as shown in Figure 30a), has a pentagonal patch on top. There are six hexagonal patches adjacent to this pentagonal patch. We consider a *layering scheme*, where the node corresponding to pentagonal patch on top is in Layer 1 (L1), the six nodes corresponding to six hexagonal patches adjacent to the pentagonal patch on top are in Layer 2 (L2) and so on. Following this layering scheme, all 32 nodes can be assigned to six layers, L1 through L6, as shown in Figure 30b. In this scheme, one node is assigned to L1, five nodes to L2, 10 nodes to L3, 10 nodes to L4, five nodes to L5, and one node to L6. There is only one pentagonal node in layers L1 and L6 and we refer to these two nodes as $P_{1,1}$ and $P_{6,1}$ respectively. There are five hexagonal nodes in layers L2 and L5 and we refer to these nodes as $H_{2,i}, 1 \leq i \leq 5$ and $H_{5,i}, 1 \leq i \leq 5$ respectively. There are five hexagonal and five pentagonal nodes in layers L3 and L4 and we refer to these nodes as $H_{i,j}, i = 3, 4, 1 \leq j \leq 5$ and $P_{i,j}, i = 3, 4, 1 \leq j \leq 5$ respectively.

The vertex set V of the SBG, is divided into two subsets, a P (for Pentagon) and H (for Hexagon), with 12 and 20 members respectively. It may be noted from Figure 30b, that P-type nodes appear only on layers 1, 3, 4 and 6 and H-type nodes appear only on layers 2, 3, 4 and 5. The SBG $G = (V, E) = ((P \cup H), E)$ is formally defined as follows:

$P = \{P_{1,1}\} \cup \{P_{i,j}, 3 \leq i \leq 4, 1 \leq j \leq 5\} \cup \{P_{6,1}\}$ and $H = \{H_{i,j}, 2 \leq i \leq 5, 1 \leq j \leq 5\}$

The edge set E is divided into 17 subsets, i.e., $E = \cup_{i=1}^{17} E_i$. Each subset is defined in Table 33.

With the formal definition of the SBG complete, it may be observed that the problem of determining the fewest number of satellites necessary to uniquely identify the region (among 32 regions) where a significant event has taken place is equivalent to computation of the Minimum Identifying Code Set problem for the SBG. Recall that, utilizing our previously defined GCS approach, suppose that the node set V' is an ICS of a graph $G = (V, E)$ and $|V'| = p$. In this case if p distinct colors are

SBG Edge Construction
$E_1 = \{(P_{1,1}, H_{2,j}), 1 \leq j \leq 5\}$
$E_2 = \{(P_{6,1}, H_{5,j}), 1 \leq j \leq 5\}$
$E_3 = \{(H_{i,j}, H_{i,(j+1) mod\ 5}), i = 2, i = 5, 1 \leq j \leq 5\}$
$E_4 = \{(H_{i,j}, P_{i,j}), i = 3, 1 \leq j \leq 5\}$
$E_5 = \{(P_{i,j}, H_{i,(j+1) mod\ 5}), i = 3, 1 \leq j \leq 5\}$
$E_6 = \{(H_{i,j}, P_{i,(j+1) mod\ 5}), i = 4, 1 \leq j \leq 5\}$
$E_7 = \{(P_{i,j}, H_{i,j}), i = 4, 1 \leq j \leq 5\}$
$E_8 = \{(H_{2,j}, H_{3,j}), 1 \leq j \leq 5\}$
$E_9 = \{(H_{2,j}, P_{3,(j-1) mod\ 5}), 1 \leq j \leq 5\}$
$E_{10} = \{(H_{2,j}, P_{3,j}), 1 \leq j \leq 5\}$
$E_{11} = \{(H_{3,j}, P_{4,j}), 1 \leq j \leq 5\}$
$E_{12} = \{(H_{3,j}, H_{4,(j-1) mod\ 5}), 1 \leq j \leq 5\}$
$E_{13} = \{(H_{3,j}, H_{4,j}), 1 \leq j \leq 5\}$
$E_{14} = \{(P_{3,j}, H_{4,j}), 1 \leq j \leq 5\}$
$E_{15} = \{(H_{4,j}, H_{5,j}), 1 \leq j \leq 5\}$
$E_{16} = \{(P_{4,j}, H_{5,j}), 1 \leq j \leq 5\}$
$E_{17} = \{(P_{4,j}, H_{5,(j-1) mod\ 5}), 1 \leq j \leq 5\}$

Table 33. Color assignment at nodes after seepage for Class IV ICS

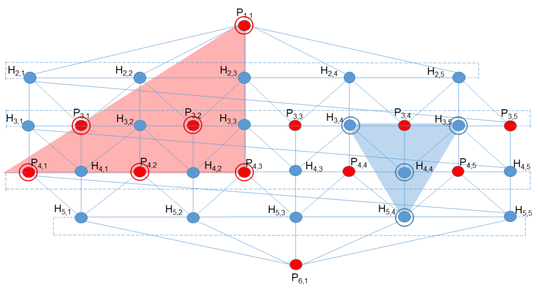
injected to V' (one distinct atomic color to one node of V'), then as by the definition of ICS for all $v \in V$, if $N^+(v) \cap V'$ is unique, all nodes of $G = (V, E)$ will have a unique color (either atomic or composite). Thus computation of MICS is equivalent to solving the GCS problem.

7.2.2 Upper Bound of MICS of SBG

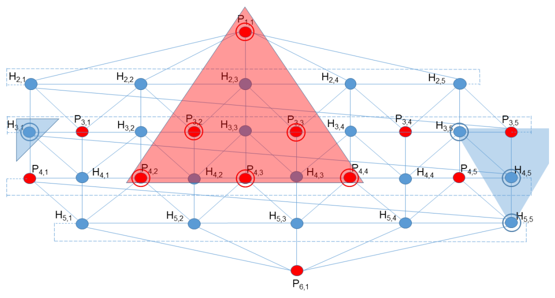
In this subsection, we first show that MICS of the SBG is at most 10 and there exists at least 26 ICS of size 10.

Theorem 13. *The MICS of SBG is at most 10.*

Proof. Inject colors A, B, C, D, E to the nodes $H_{2,j}, 1 \leq j \leq 5$ and colors E, F, G, H, I, J to the nodes $H_{5,j}, 1 \leq j \leq 5$. Injection of 10 different colors at


$P_{1,1}$
$H_{2,1}$ $H_{2,2}$ $H_{2,3}$ $H_{2,4}$ $H_{2,5}$
$H_{3,1}$ $P_{3,1}$ $H_{3,2}$ $P_{3,2}$ $H_{3,3}$ $P_{3,3}$ $H_{3,4}$ $P_{3,4}$ $H_{3,5}$ $P_{3,5}$
$P_{4,1}$ $H_{4,1}$ $P_{4,2}$ $H_{4,2}$ $P_{4,3}$ $H_{4,3}$ $P_{4,4}$ $H_{4,4}$ $P_{4,5}$ $H_{4,5}$
$H_{5,1}$ $H_{5,2}$ $H_{5,3}$ $H_{5,4}$ $H_{5,5}$
$P_{6,1}$

(a) Class IIA Motif Assignment I


$P_{1,1}$
$H_{2,1}$ $H_{2,2}$ $H_{2,3}$ $H_{2,4}$ $H_{2,5}$
$H_{3,1}$ $P_{3,1}$ $H_{3,2}$ $P_{3,2}$ $H_{3,3}$ $P_{3,3}$ $H_{3,4}$ $P_{3,4}$ $H_{3,5}$ $P_{3,5}$
$P_{4,1}$ $H_{4,1}$ $P_{4,2}$ $H_{4,2}$ $P_{4,3}$ $H_{4,3}$ $P_{4,4}$ $H_{4,4}$ $P_{4,5}$ $H_{4,5}$
$H_{5,1}$ $H_{5,2}$ $H_{5,3}$ $H_{5,4}$ $H_{5,5}$
$P_{6,1}$

(b) Class IIA Motif Assignment II (Assignment I Shifted one Position to the Right)

Figure 31. Examples of Color Assignments using Motif IIA

these 10 nodes, will cause color seepage to all other nodes of SBG. The color seepage will be constrained by the topological structure of the SBG. It may be verified that because of the constraint imposed by the SBG structure, and the fact that seepage takes place only to the neighbors of the node where a color is injected, the 32 nodes of the SBG will have the color assignment shown in Table 34. In the entries of Table 34, $H_{2,1} : A^*BE$ implies that the color A was *injected* at the node $H_{2,1}$ and the colors B and E *seeped* into the node $H_{2,1}$, from the adjacent nodes $H_{2,2}$ and $H_{2,5}$, where the colors B and E were injected. In general, if an alphabet A through E (representing distinct colors), appears *with* a * as a part of a string attached to a node (such as $H_{2,1}$), it implies that the color was *injected* at that node. On the other hand, if an alphabet appears *without* a * as a part of a string attached to a node, it implies that the color *seeped* into that node from one of the adjacent nodes. It may be verified that the color assignment to the nodes, as shown in Table 34 is *unique* (i.e, no two nodes have the same color or *strings* assigned to them). $\square$

Theorem 14. *At least 26 distinct size-10 ICSs of the SBG exist.*

Node: Color	Node: Color	Node: Color	Node: Color
$P_{1,1}$: $ABCDE$	$H_{2,1}$: A^*BE	$H_{2,2}$: AB^*C	$H_{2,3}$: BC^*D
$H_{2,4}$: CD^*E	$H_{2,5}$: DE^*A	$H_{3,1}$: A	$P_{3,1}$: AB
$H_{3,2}$: B	$P_{3,2}$: BC	$H_{3,3}$: C	$P_{3,3}$: CD
$H_{3,4}$: D	$P_{3,4}$: DE	$H_{3,5}$: E	$P_{3,5}$: AE
$P_{4,1}$: JF	$H_{4,1}$: F	$P_{4,2}$: FG	$H_{4,2}$: G
$P_{4,3}$: GH	$H_{4,3}$: H	$P_{4,4}$: HI	$H_{4,4}$: I
$P_{4,5}$: IJ	$H_{4,5}$: J	$H_{5,1}$: JF^*G	$H_{5,2}$: FG^*H
$H_{5,3}$: GH^*I	$H_{5,4}$: HI^*J	$H_{5,5}$: IJ^*F	$P_{6,1}$: $FGHIJ$

Table 34. Color assignment at nodes after seepage in the SBG

Proof. The 26 different ways in which 10 colors can be injected into 10 nodes of the SBG such that every node of the SBG receives a unique color can be divided into four classes.

- *Class I:* Inject colors A, B, C, D, E to the nodes $H_{2,j}, 1 \leq j \leq 5$ and colors E, F, G, H, I, J to the nodes $H_{5,j}, 1 \leq j \leq 5$. As shown in Table 34, such an injection ensures that each of the 32 nodes of the SBG receives a unique color. It may be observed that the node set where the colors are injected in this Class all have degree six, corresponding to hexagonal patches on the surface of the soccer ball. Only one ICS of the 26, belongs to Class I.

- *Class II:* The node set where the colors injected are in this Class is made up of six nodes of degree five (corresponding to the pentagonal patches of the soccer ball) and four nodes of degree six (corresponding to the hexagonal patches of the soccer ball). This Class can be subdivided into two sub-classes and we will refer to them as *Class II-A* and *Class II-B* respectively. As seen in Figure 30b, the SBG graph is somewhat symmetric in the sense that the layers 4, 5 and 6 are close to being mirror images of layers 1, 2 and 3. Because of this symmetry, the Class II-A color injections are

Node	Color
$P_{1,1}$	$P^c_{1,1}$
$H_{2,j}$	$P^c_{1,1}P^c_{3,j}$
$H_{2,(j+1)mod\ 5}$	$P^c_{1,1}P^c_{3,j}P^c_{3,(j+1)mod\ 5}$
$H_{2,(j+2)mod\ 5}$	$P^c_{1,1}P^c_{3,(j+1)mod\ 5}$
$H_{2,(j+3)mod\ 5}$	$P^c_{1,1}H^c_{3,(j+3)mod\ 5}$
$H_{2,(j+4)mod\ 5}$	$P^c_{1,1}H^c_{3,(j+4)mod\ 5}$
$H_{3,j}$	$P^c_{3,j}P^c_{4,j}$
$P_{3,j}$	$P^c_{3,j}$
$H_{3,(j+1)mod\ 5}$	$P^c_{3,j}P^c_{3,(j+1)mod\ 5}P^c_{4,(j+1)mod\ 5}$
$P_{3,(j+1)mod\ 5}$	$P^c_{3,(j+1)mod\ 5}$
$H_{3,(j+2)mod\ 5}$	$P^c_{3,(j+1)mod\ 5}P^c_{4,(j+2)mod\ 5}$
$P_{3,(j+2)mod\ 5}$	$H^c_{3,(j+3)mod\ 5}$
$H_{3,(j+3)mod\ 5}$	$H^c_{3,(j+3)mod\ 5}H^c_{4,(j+3)mod\ 5}$
$P_{3,(j+3)mod\ 5}$	$H^c_{3,(j+3)mod\ 5}H^c_{4,(j+3)mod\ 5}H^c_{3,(j+4)mod\ 5}$
$H_{3,(j+4)mod\ 5}$	$H^c_{3,(j+4)mod\ 5}H^c_{4,(j+3)mod\ 5}$
$P_{3,(j+4)mod\ 5}$	$H^c_{4,(j+4)mod\ 5}$
$P_{4,j}$	$P^c_{4,j}$
$H_{4,j}$	$P^c_{3,j}P^c_{4,j}P^c_{4,(j+1)mod\ 5}$
$P_{4,(j+1)mod\ 5}$	$P^c_{4,(j+1)mod\ 5}$
$H_{4,(j+1)mod\ 5}$	$P^c_{3,(j+1)mod\ 5}P^c_{4,(j+1)mod\ 5}P^c_{4,(j+2)mod\ 5}$
$P_{4,(j+2)mod\ 5}$	$P^c_{4,(j+2)mod\ 5}$
$H_{4,(j+2)mod\ 5}$	$P^c_{4,(j+2)mod\ 5}H^c_{3,(j+3)mod\ 5}$
$P_{4,(j+3)mod\ 5}$	$H^c_{3,(j+3)mod\ 5}H^c_{4,(j+3)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$H_{4,(j+3)mod\ 5}$	$H^c_{4,(j+3)mod\ 5}H^c_{3,(j+3)mod\ 5}$
	$H^c_{3,(j+4)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$P_{4,(j+4)mod\ 5}$	$H^c_{4,(j+4)mod\ 5}H^c_{3,(j+3)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$H_{4,(j+4)mod\ 5}$	$P^c_{4,j}H^c_{4,(j+4)mod\ 5}$
$H_{5,j}$	$P^c_{4,j}P^c_{4,(j+1)mod\ 5}$
$H_{5,(j+1)mod\ 5}$	$P^c_{4,(j+1)mod\ 5}P^c_{4,(j+2)mod\ 5}$
$H_{5,(j+2)mod\ 5}$	$P^c_{4,(j+2)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$H_{5,(j+3)mod\ 5}$	$H^c_{5,(j+3)mod\ 5}H^c_{4,(j+3)mod\ 5}$
$H_{5,(j+4)mod\ 5}$	$P^c_{4,j}H^c_{5,(j+3)mod\ 5}$
$P_{6,1}$	$H^c_{5,(j+3)mod\ 5}$

Table 35. Color assignment at nodes after seepage for Class II ICS

mirror images of the Class II-B color injection. Accordingly, in this section we will focus our discussion primarily on Class II-A, as color injection for class II-B be can be obtained easily from color injection in Class II-A. We introduce the notion of a *motif*, and by motif we imply a set of either P-type (degree five) or H-type (degree six) nodes. It will be clear from further discussion that the Class II-A solutions comprise of one P-type motif and one H-type motif. These two motifs *complement* each other to produce a solution together. The motif-pairs can be *slid* along the structure of the SBG to produce a set of five solutions that make up the Class II-A. The five solutions that make up the Class II-B can be constructed in a similar fashion.

For the ICS that belong to Class II, the P-type motif is made up of the set of six nodes $\{P_{1,1}, P_{3,j}, P_{3,(j+1)mod\ 5}, P_{4,j}, P_{4,(j+1)mod\ 5}, P_{4,(j+2)mod\ 5}\}$. The H-type motif that complements the P-type motif is made up of the set of four nodes $\{H_{3,(j+3)mod\ 5}, H_{3,(j+4)mod\ 5}, H_{4,(j+3)mod\ 5}, H_{5,(j+3)mod\ 5}\}$. One complete solution (i.e., ICS) is obtained by choosing a value of $j, 1 \leq j \leq 5$. The Figure 31a and Figure 31b show the solutions with $j = 1$ and $j = 2$ respectively. As shown in Figure 31, changing the index j from 1 to 2, has the effect of sliding the motif along the structure of the SBG. By changing j from 1 through 5 (i.e., sliding the motif 5 times), 5 different ICS can be computed. The colors that will be associated with the nodes of the SBG, if they are injected at the motif nodes, are shown in Table 35. The first column of the table indicates the node and the second column provides the color assigned to that node. For example, in row 3 of Table 35, the node $H_{2,(j+1)mod\ 5}$ receives the colors injected at motif nodes $P_{1,1}, P_{3,j}, P_{3,(j+1)mod\ 5}$ and is denoted by $P^c_{1,1}P^c_{3,j}P^c_{3,(j+1)mod\ 5}$. It may be verified that every node of the SBG has a *color associated with it and no two nodes have the same color assignment.*

- *Class III:* As in Class II, the Class III ICS is made up of six nodes of degree five

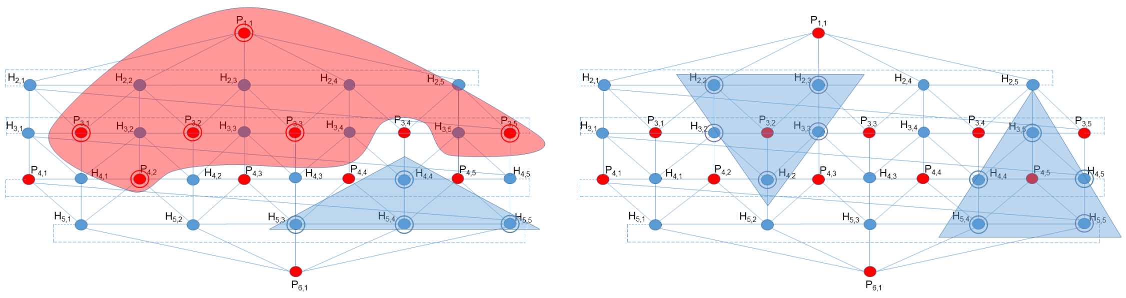

(a) Class IIIA Motif Assignment (b) Class IV Motif Assignment

Figure 32. Examples of Color Assignments using Motifs IIIA and IV

and four nodes of degree six. Moreover, this Class also can be subdivided into two sub-classes and we will refer to them as Class III-A and Class III-B respectively. In this section we will restrict our discussion on Class III-A, as color injection for Class III-B can be obtained as a mirror image of Class III-A. It will be clear from further discussion that, as in Class II, the Class III solutions also comprise of one P-type motif and one H-type motif and they complement each other to produce a solution together. As in Class II, the motif-pairs can be slid along the structure of the SBG to produce a set of five solutions that make up the Class III-A. The five solutions that make up the Class III-B can be constructed in a similar fashion.

The P-type motif is made up of the set of six nodes $\{P_{1,1}, P_{3,j}, P_{3,(j+1)mod\ 5}, P_{3,(j+2)mod\ 5}, P_{3,(j+4)mod\ 5}, P_{4,(j+1)mod\ 5}\}$. As shown in Figure 32a, changing the index j from 1 to 5, has the effect of sliding the motif along the structure of the SBG. The H-type motif that complements the P-type motif is made up of the set of four nodes $\{H_{4,(j+3)mod\ 5}, H_{5,(j+2)mod\ 5}, H_{5,(j+3)mod\ 5}, H_{5,(j+4)mod\ 5}\}$. One complete solution is obtained by choosing a value of $j, 1 \leq j \leq 5$. By moving the P-type and H-type motifs in tandem by changing the value of the index from 1 to 5, five different solutions can be obtained. The colors that will be associated with the nodes of the SBG, if the

Node	Color
$P_{1,1}$	$P^c_{1,1}$
$H_{2,j}$	$P^c_{1,1}P^c_{3,j}P^c_{3,(j+4)mod\ 5}$
$H_{2,(j+1)mod\ 5}$	$P^c_{1,1}P^c_{3,j}P^c_{3,(j+1)mod\ 5}$
$H_{2,(j+2)mod\ 5}$	$P^c_{1,1}P^c_{3,(j+1)mod\ 5}P^c_{3,(j+2)mod\ 5}$
$H_{2,(j+3)mod\ 5}$	$P^c_{1,1}P^c_{3,(j+2)mod\ 5}$
$H_{2,(j+4)mod\ 5}$	$P^c_{1,1}P^c_{3,(j+4)mod\ 5}$
$H_{3,j}$	$P^c_{3,j}P^c_{3,(j+4)mod\ 5}$
$P_{3,j}$	$P^c_{3,j}$
$H_{3,(j+1)mod\ 5}$	$P^c_{3,j}P^c_{3,(j+1)mod\ 5}P^c_{4,(j+1)mod\ 5}$
$P_{3,(j+1)mod\ 5}$	$P^c_{3,(j+1)mod\ 5}$
$H_{3,(j+2)mod\ 5}$	$P^c_{3,(j+1)mod\ 5}P^c_{3,(j+2)mod\ 5}$
$P_{3,(j+2)mod\ 5}$	$P^c_{3,(j+2)mod\ 5}$
$H_{3,(j+3)mod\ 5}$	$P^c_{3,(j+2)mod\ 5}H^c_{4,(j+3)mod\ 5}$
$P_{3,(j+3)mod\ 5}$	$H^c_{4,(j+3)mod\ 5}$
$H_{3,(j+4)mod\ 5}$	$H^c_{4,(j+3)mod\ 5}P^c_{3,(j+4)mod\ 5}$
$P_{3,(j+4)mod\ 5}$	$P^c_{3,(j+4)mod\ 5}$
$P_{4,j}$	$H^c_{5,(j+4)mod\ 5}$
$H_{4,j}$	$P^c_{4,(j+1)mod\ 5}P^c_{3,j}$
$P_{4,(j+1)mod\ 5}$	$P^c_{4,(j+1)mod\ 5}$
$H_{4,(j+1)mod\ 5}$	$P^c_{4,(j+1)mod\ 5}P^c_{3,(j+1)mod\ 5}$
$P_{4,(j+2)mod\ 5}$	$H^c_{5,(j+2)mod\ 5}$
$H_{4,(j+2)mod\ 5}$	$P^c_{3,(j+2)mod\ 5}H^c_{5,(j+2)mod\ 5}$
$P_{4,(j+3)mod\ 5}$	$H^c_{4,(j+3)mod\ 5}H^c_{5,(j+2)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$H_{4,(j+3)mod\ 5}$	$H^c_{4,(j+3)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$P_{4,(j+4)mod\ 5}$	$H^c_{4,(j+3)mod\ 5}H^c_{5,(j+3)mod\ 5}H^c_{5,(j+4)mod\ 5}$
$H_{4,(j+4)mod\ 5}$	$P^c_{3,(j+4)mod\ 5}H^c_{5,(j+4)mod\ 5}$
$H_{5,j}$	$H^c_{5,(j+4)mod\ 5}P^c_{4,(j+1)mod\ 5}$
$H_{5,(j+1)mod\ 5}$	$H^c_{5,(j+2)mod\ 5}P^c_{4,(j+1)mod\ 5}$
$H_{5,(j+2)mod\ 5}$	$H^c_{5,(j+2)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$H_{5,(j+3)mod\ 5}$	$H^c_{5,(j+3)mod\ 5}H^c_{5,(j+4)mod\ 5}$
	$H^c_{5,(j+2)mod\ 5}H^c_{4,(j+3)mod\ 5}$
$H_{5,(j+4)mod\ 5}$	$H^c_{5,(j+4)mod\ 5}H^c_{5,(j+3)mod\ 5}$
$P_{6,1}$	$H^c_{5,(j+2)mod\ 5}H^c_{5,(j+3)mod\ 5}H^c_{5,(j+4)mod\ 5}$

Table 36. Color assignment at nodes after seepage for Class III ICS

they are injected at the motif nodes, are shown in Table 36. As seen earlier in Class II, it may be verified that every node of the SBG has a *color associated with it and no two nodes have the same color assignment.*

- *Class IV:* As in Class I, the Class IV ICS are made up of 10 nodes of degree six (i.e., the nodes corresponding to hexagonal patches). This Class comprises of five ICS and cannot be subdivided like in Classes II and III.

This class comprises of two H-type motifs made up of five hexagonal nodes each. The first motif comprises of $\{H_{2,(j+1) mod\ 5}, H_{2,(j+2) mod\ 5}, H_{3,(j+1) mod\ 5}, H_{3,(j+2) mod\ 5}, H_{4,(j+1) mod\ 5}\}$. The other motif comprises of $\{H_{3,(j+4) mod\ 5}, H_{4,(j+3) mod\ 5}, H_{4,(j+4) mod\ 5}, H_{5,(j+3) mod\ 5}, H_{5,(j+4) mod\ 5}\}$. As shown in Figure 32b, changing the index j from 1 to 5, has the effect of sliding the motif along the structure of the SBG. One complete solution is obtained by choosing a value of $j, 1 \leq j \leq 5$. By moving two H-type motifs in tandem, changing the value of the index from 1 to 5, five different solutions can be obtained. As in Classes II and III, if colors are injected at the motif nodes, then every node of the SBG will have a *color associated with it and no two nodes will have the same color assignment.* Due to similarity, we omit the corresponding color assignment table.

This concludes proof of Theorem 14. □

7.2.3 Lower Bound of MICS of SBG

In Figure 30b, we have provided a *layered* representation of the SBG, where 32 nodes of the SBG are placed in six layers, indicated by L1 through L6. The layers L1 through L3 constitute the *top half* of the SBG and the layers L4 through L6 constitute the *bottom half*. As the two halves are symmetric, similar argument can be applied to both of them.

Lemma 15. *An MICS must select at least* 4 *nodes from each half. In other words, at least* 4 *distinct colors need to be injected in each half.*

Proof. We provide arguments for the top half of the SBG, and we first show that three distinct colors are necessary to ensure that each node in top half receives a distinct color (either through injection, seepage or combination of the two). Let's consider layers L1 and L2. No matter which nodes in bottom half are injected with colors, these colors will not seep into the nodes in L1 and L2 (colors seep only to adjacent nodes), coloring in bottom half nodes will not affect the colors associated with nodes in L1 or L2. Since L1 and L2 have six nodes, six distinct colors need to be associated with them. It can be easily verified that in the SBG, in order to ensure distinct colors to each one of the six nodes in L1 and L2 at least three colors must be injected to three nodes in the top half of the SBG. Clearly at least three nodes must be selected from each half so that nodes in L1, L2, L5 and L6 receive distinct colors. With injection of three colors, up to $2^3 - 1 = 7$ colors (excluding an empty combination) can be generated.

Next we show that three colors are not sufficient to color L1 and L2. WLOG, we use alphabets $\{A, B, C\}$ to represent three colors. As mentioned above, 7 distinct colors can be generated with these three colors (three primary and four composite), $\{A, B, C, AB, AC, BC, ABC\}$. Simple counting shows that each alphabet (color) appears exactly 4 times. Suppose there is a proper injection using A, B, C that ensures all nodes in L1 and L2 received distinct colors. Since seven distinct colors can be generated with three primary colors, and L1 and L2 has only six nodes, it implies that one of the seven colors (primary or composite) is not used while coloring the nodes of L1 and L2. This implies that at least one of the alphabets A, B, C is appearing three times instead of four in the alphabet strings (representing colors) associated with the nodes of L1 and L2. WLOG *we assume that color A is appearing 3 times.* There are four possible locations for injection of color A in the top half of the SBG. In the following, we examine them all.

1. A is injected on L1, i.e., at $P_{1,1}$. A would then appear at all nodes in L1 and L2, making its appearance six times, contradicting the assumption.
2. A is injected on L2, i.e., one of $H_{2,i}(1 \leq i \leq 5)$ nodes. Thus, A appears four times (three nodes in L2 and one node in L1) contradicting the assumption.
3. A is injected on one of the hexagonal nodes L3, i.e., one of the $H_{3,i}, 1 \leq i \leq 5$. Since $H_{3,i}$ has only one neighbor in on L1 and L2 ($H_{2,i}$), in this case A will appear only on one node in L2, making its appearance one time, contradicting the assumption.
4. A is injected on one of the pentagonal nodes L3, i.e., one of the $P_{3,i}, 1 \leq i \leq 5$. Since $P_{3,i}$ has only two neighbors in on L1 and L2 ($H_{2,(i-1) mod\ 5}$ and $H_{2,i}$), in this case A will appear only on two nodes in L2, making its appearance two times, contradicting the assumption.

As there is no location for injection of A, we can conclude that 3 colors are inadequate to ensure that all nodes in L1 and L2 receive a unique color. Similar arguments can be made for coloring of nodes in L5 and L6. Therefore the lower bound of MICS for the SBG must be at least $4 + 4 = 8$. □

Lemma 16. *MICS of the SBG is at least 9.*

Proof. In GCS problem (which is equivalent to the MICS problem), each node is assigned a color, which may be a primary or a composite color. A primary color is indicated by one alphabet and a composite color by a *string* of alphabets. The number of alphabets that appear in a string determines the *length* of that string. We establish the lemma by providing arguments based on the sum of the length of strings associated with each one of the 32 nodes of the SBG. We will refer to the sum of the length of strings associated with each one of the 32 nodes of the SBG as "total string length".

We use the term "valid injection" to imply an injection of colors to the nodes that ensures that all 32 nodes of the SBG receive a distinct color. Suppose there exists more than one valid injections using eight colors. Among the set of all valid injections, we consider the one whose total string length is minimum. The lower bound of the total string length for a valid injection with eight colors is 56. This is true, as with eight injected colors, at most eight nodes of the SBG can have associated strings of length one, and the remaining 24 nodes must have strings of length at least two. Thus the lower bound on the string length must be $8 \times 1 + 24 \times 2 = 56$. The upper bound on the total string length with injection of at most eight colors is also 56. This is true for the following reason. If a color is injected on a hexagonal node, then it will appear seven times (six neighbors and the node itself). Similarly for a pentagonal node, the color will appear six times. Therefore, the upper bound of total string length is $7 \times 8 = 56$. It may be noted that the total string length is 56 if and only if all colors are injected on hexagonal nodes. However, it is impossible to achieve a valid injection by injecting eight colors only on hexagonal nodes. Consider the top half of the SBG. In order to color nodes on L1, at least one color, say A, must be injected on one node on L2. WLOG, we assume that A is injected on $H_{2,i}, 1 \leq i \leq 5$. We consider two scenarios:

1. No other color is injected at the nodes on L2. In this case, the other colors are

injected at three hexagonal nodes on L3. Because of injection of A at $H_{2,i}$, after seepage, all six adjacent nodes, $P_{1,1}, H_{2,(i-1)mod\ 5}, H_{2,(i+1)mod\ 5}, H_{3,(i-1)mod\ 5}, P_{3,i}$, $H_{3,i}$, will have color A. In order ensure that all these nodes have distinct colors, colors must be injected on $H_{3,(i-1)mod\ 5}, H_{3,i}, H_{3,(i+1)mod\ 5}$. However, if such an injection is made, the nodes $H_{2,(i+2)mod\ 5}$ and $H_{2,(i+3)mod\ 5}$ will not receive any color. Accordingly, such an injection will not be a valid injection.

2. One or more colors are injected at the nodes on L2. Suppose a different color B is injected at a node different from $H_{2,i}$ (where color A is injected). Due to the SBG topology, no matter which node on L2 is injected with B, one node on L2 and the node on L1 must have color AB after seepage. In order to ensure distinct colors on these two nodes, a third color C must be injected on another node. After injection of C, one of the two nodes that had the color AB before injection of C, will have the color AB and the other will have ABC. However, if one node has a string of length three, the lower bound of the total string length can longer be 56. It has to be at least 57, thus exceeding the upper bound (56), that is possible with injection of at most eight colors.

□

Theorem 17. *The lower bound of MICS of the SBG is at least nine. In other words, eight colors are insufficient to ensure that all nodes of the the SBG receives a distinct color.*

Proof. Follows from Lemmas 15 and 16. □

7.3 Conclusion

In this chapter, we studied niche Identifying Code based sensor placement problems for restricted graph classes, in one and two dimensional spaces for accurate identification of fault locations, as well as monitoring regions of the earth for anomalous events. For one and two dimensional problems, we studied the concept of Identifying Codes in a specialized class of graphs - Unit Interval Bigraphs and Unit Dics Bigraphs. We presented a dynamic programming based approach to determine the optimal solution in the one dimensional scenario. To the best of our knowledge, this is the first polynomial time algorithm presented for the computation of MDCS for Unit Interval Bigraphs, when their interval representations are given. Additionally, we presented an Integer Linear Program for the two dimensional scenario, namely for the Unit Disc Bigraphs.

Furthermore, we studied an event monitoring problem with satellites as sensors and a soccer ball as a model of the planet Earth. We provided upper and lower bound of the MICS problem where the difference between the bounds is just one, implying that our solution is close to being optimal. It must be noted that, the regions (patches) on the soccer ball are of regular shapes. In our future work, we plan to study regions with irregular shapes, thereby relaxing the hexagonal and pentagonal region constraints.

Chapter 8

CONCLUSION

For all the uncertainty in the world, one thing is certain for sure - anomalies manifest, in some form or the other, in complex networks. Therefore, it is fundamental to not only design approaches which can quickly detect the presence of anomalies in the network, but also, to identify the source(s) of anomalies in the network, so that preventive measures can be implemented. Most of the approaches described in the literature, deploy sensors in the complex network following the Set Cover approach. Over the course of this thesis, we have shown how such approaches fail to *uniquely identify* the source(s) of anomaly, and how the notion of Identifying Codes can be effectively utilized to uniquely monitor the complex network.

We journeyed across four major domains namely, human-human interaction networks, critical infrastructure networks, online social networks and water distribution networks, in regards to the problems addressed in this thesis. On this journey, we have not only shown the effectiveness of Identifying Codes but also introduced highly relevant problems, which had not been studied before, most notably, the Budget Constrained Identifying Code problem presented in chapter 6, the scalable Identifying Code problem presented in chapter 5, the Minimum Discriminating Code Set problem in the presence of adversary in chapter 4, etc. The objective of this dissertation was to bring attention to the idea of Identifying Codes and impress upon the research community its applicability in the monitoring domain. That being said however, following are some of the directions which may be examined and further pursued -

- *Probabilistic Identifying Codes*: As mentioned previously, one key assumption made in our problem setup is that, when an individual (or a node) engages in anomalous behavior, all of its neighbors become aware of it. Generally speaking, this is a purely deterministic setting and may not be a valid assumption in the real world. A probabilistic setting may be even more appropriate, where there is a probability value associated with each edge in the network. These probability values would indicate the probabilities of the signal (anomalous behavior) reaching the neighbors of the node engaging in anomalous behavior. In other words, the probabilistic setting is a more generalized version of the deterministic setting. There has been some initial works on probabilistic Identifying Codes, but these have mostly focused on random graphs (Frieze et al. 2007). In our work, the graphs obtained from the respective domains, may not follow the characteristics associated with random graphs. Therefore, a modification to the definition of Identifying Codes is required as well as the development of approaches to determine the minimum cardinality Identifying Code set.

- *Generalization of Multiple Simultaneous Failures*: In chapter 3, we introduced a novel concept where two nodes could simultaneously become active in anomalous behavior. For our approach to be applicable to the real world setting, it would be interesting to develop algorithms for the scenario where k multiple simultaneously activations can be effectively handled. In other words, this implies a need for the development of a generalized approach to detect multiple simultaneous activations as opposed to the much more restrictive approach presented in this chapter.
- *Budgeted Identifying and Discriminating Codes*: Although we presented an integer linear program to determine the optimal solution to the Budgeted Identifying Code problem in chapter 6 and the guarantee of non-existence of

constant factor approximation algorithm, we left open the possibility of the existence of a logarithmic factor approximation algorithm to this problem. Additionally, the budgeted scenario was not addressed for the Discriminating Code problem, discussed in chapter 4.

- *Deep Learning for Identifying Codes*: Over the past half decade, there has been significant development of deep learning architectures to solve graph problems. Architectures such as Graph Neural Networks (GNN) and Graph Convolutional Neural Networks (GCNN) have been utilized to solve combinatorial graph problems (Sato, Yamada, and Kashima 2019; Cappart et al. 2021). Furthermore, (Sato, Yamada, and Kashima 2019) have even derived approximation bounds for GNN based approaches for problems such as Dominating Set and Vertex Cover. With more architectures focusing on solving combinatorial problems coupled with the prowess of Graphical Processing Units (GPUs) and Tensor Processing Units (TPUs), it is entirely feasible to model an architecture for the Identifying Code problem, thereby utilizing GPUs and TPUs to solve larger problem instances.

As a final remark in this dissertation, we would like to impress upon the readers, the universality of our developed approaches. Our algorithms *only* focus on the placement and not the actual development/manufacturing of the sensors. As evident from the multiple application domains, by simply changing the type of the sensor, a complex network operator can utilize our approach in another domain. For instance, by changing the sensor type from law enforcement agents to PMUs, one may utilize our approach in the critical infrastructure domain from the human-human interaction domain.

REFERENCES

Adam, George A, Petr Smirnov, Anna Goldenberg, David Duvenaud, and Benjamin Haibe-Kains. 2018. "Stochastic Combinatorial Ensembles for Defending Against Adversarial Examples." *arXiv:1808.06645.*

Anthopoulos, Leonidas G. 2017. "The smart city in practice." In *Understanding Smart Cities: A Tool for Smart Government or an Industrial Trick?,* 47–185. Springer.

Banerjee, Joydeep, Kaustav Basu, and Arunabha Sen. 2018a. "Analysing robustness in intra-dependent and inter-dependent networks using a new model of interdependency." *International Journal of Critical Infrastructures* 14 (2): 156–181.

———. 2018b. "On hardening problems in critical infrastructure systems." *International Journal of Critical Infrastructure Protection* 23:49–67.

Banerjee, Joydeep, Anamitra Pal, Kaustav Basu, Malhar Padhee, and Arunabha Sen. 2017. "Finding K Contingency List in Power Networks using a New Model of Dependency." *arXiv preprint arXiv:1705.07410.*

Basu, Kaustav. 2019. "Identification of the Source (s) of Misinformation Propagation Utilizing Identifying Codes." In *Companion Proceedings of The 2019 World Wide Web Conference,* 7–11.

Basu, Kaustav, Sandipan Choudhuri, Arunabha Sen, and Aniket Majumdar. 2018. "Insights from statistical analysis of opioid data." *arXiv preprint arXiv:1805.05509.*

Basu, Kaustav, Sanjana Dey, Subhas Nandy, and Arunabha Sen. 2019. "Sensor Networks for Structural Health Monitoring of Critical Infrastructures Using Identifying Codes." In *2019 15th International Conference on the Design of Reliable Communication Networks (DRCN),* 43–50. IEEE.

Basu, Kaustav, Malhar Padhee, Sohini Roy, Anamitra Pal, Arunabha Sen, Matthew Rhodes, and Brian Keel. 2018. "Health Monitoring of Critical Power System Equipments Using Identifying Codes." In *International Conference on Critical Information Infrastructures Security,* 29–41. Springer.

Basu, Kaustav, and Arunabha Sen. 2019a. "Monitoring individuals in drug trafficking organizations: a social network analysis." In *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM),* 480–483. IEEE.

Basu, Kaustav, and Arunabha Sen. 2019b. “On augmented identifying codes for monitoring drug trafficking organizations.” In *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM),* 1136–1139. IEEE.

———. 2021a. “Epidemiological Model Independent Misinformation Source Identification.”

———. 2021b. “Identifying individuals associated with organized criminal networks: a social network analysis.” *Social Networks* 64:42–54.

———. 2022. “Sensor Network Design for Uniquely Identifying Sources of Contamination in Water Distribution Networks.”

Basu, Kaustav, Chenyang Zhou, Arunabha Sen, and Victoria Horan Goliber. 2018. “A novel graph analytic approach to monitor terrorist networks.” In *2018 IEEE International Conference on Social Computing & Networking (SocialCom),* 1159–1166. IEEE.

Becejac, Tamara, Payman Dehghanian, and Mladen Kezunovic. 2016. “Probabilistic assessment of PMU integrity for planning of periodic maintenance and testing.” In *2016 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS),* 1–6. IEEE.

Bernstein, Leandra. 2017. “From France to the U.S. terrorists ’known to authorities’ carry out deadly attacks.” https://wjla.com/news/nation-world/terrorists-known-to-authorities-carry-out-deadly-attacks.

Berzinji, Ala, Lisa Kaati, and Ahmed Rezine. 2012. “Detecting key players in terrorist networks.” In *2012 European Intelligence and Security Informatics Conference,* 297–302. IEEE.

Bhuiyan, Md Zakirul Alam, Guojun Wang, Jiannong Cao, and Jie Wu. 2014. “Sensor placement with multiple objectives for structural health monitoring.” *ACM Transactions on Sensor Networks (TOSN)* 10 (4): 1–45.

Bonacich, Phillip. 1972. “Factoring and weighting approaches to status scores and clique identification.” *Journal of mathematical sociology* 2 (1): 113–120.

Borgatti, Stephen P. 2006. “Identifying sets of key players in a social network.” *Computational & Mathematical Organization Theory* 12 (1): 21–34.

Bousquet, Nicolas, Aurélie Lagoutte, Zhentao Li, Aline Parreau, and Stéphan Thomassé. 2015. “Identifying codes in hereditary classes of graphs and VC-dimension.” *SIAM Journal on Discrete Mathematics* 29 (4): 2047–2064.

Brandes, Ulrik. 2001. “A faster algorithm for betweenness centrality.” *Journal of mathematical sociology* 25 (2): 163–177.

Bright, David A, and Jordan J Delaney. 2013. “Evolution of a drug trafficking network: Mapping changes in network structure and function across time.” *Global Crime* 14 (2-3): 238–260.

Bright, David A, Catherine Greenhill, Michael Reynolds, Alison Ritter, and Carlo Morselli. 2015. “The use of actor-level attributes and centrality measures to identify key actors: A case study of an Australian drug trafficking network.” *Journal of contemporary criminal justice* 31 (3): 262–278.

Bright, David A, Caitlin E Hughes, and Jenny Chalmers. 2012. “Illuminating dark networks: A social network analysis of an Australian drug trafficking syndicate.” *Crime, law and social change* 57 (2): 151–176.

Cappart, Quentin, Didier Chételat, Elias Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. 2021. “Combinatorial optimization and reasoning with graph neural networks.” *arXiv preprint arXiv:2102.09544.*

Cardei, Mihaela, and Jie Wu. 2004. “Coverage in wireless sensor networks.” *Handbook of sensor networks* 21:201–202.

———. 2006. “Energy-efficient coverage problems in wireless ad-hoc sensor networks.” *Computer communications* 29 (4): 413–420.

Carley, Kathleen M, Matthew Dombroski, Maksim Tsvetovat, Jeffrey Reminga, Natasha Kamneva, et al. 2003. “Destabilizing dynamic covert networks.” In *Proceedings of the 8th international Command and Control Research and Technology Symposium.* Citeseer.

Carley, Kathleen M, Ju-Sung Lee, and David Krackhardt. 2002. “Destabilizing networks.” *Connections* 24 (3): 79–92.

Carter, Kevin M, James F Riordan, and Hamed Okhravi. 2014. “A game theoretic approach to strategy determination for dynamic platform defenses.” In *Proceedings of the first ACM workshop on moving target defense,* 21–30.

Charbit, Emmanuel, Irene Charon, Gérard Cohen, and Olivier Hudry. 2006. "Discriminating codes in bipartite graphs." *Electronic Notes in Discrete Mathematics* 26:29–35.

Charbit, Emmanuel, Irène Charon, Gérard Cohen, Olivier Hudry, and Antoine Lobstein. 2008. "Discriminating codes in bipartite graphs: bounds, extremal cardinalities, complexity." *Advances in Mathematics of Communications* 2 (4): 403.

Charon, Irene, Gérard Cohen, Olivier Hudry, and Antoine Lobstein. 2008. "Discriminating codes in (bipartite) planar graphs." *European Journal of Combinatorics* 29 (5): 1353–1364.

Charon, Irene, Iiro Honkala, Olivier Hudry, and Antoine Lobstein. 2007. "Structural properties of twin-free graphs." *the electronic journal of combinatorics,* R16–R16.

Charon, Irene, Olivier Hudry, and Antoine Lobstein. 2002. "Identifying and locating-dominating codes: NP-completeness results for directed graphs." *IEEE Transactions on Information Theory* 48 (8): 2192–2200.

Charon, Irène, Olivier Hudry, and Antoine Lobstein. 2003. "Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard." *Theoretical Computer Science* 290 (3): 2109–2120.

Chatfield, Brycent, and Rami J Haddad. n.d. "Moving target defense intrusion detection system for ipv6 based smart grid advanced metering infrastructure." In *SoutheastCon 2017.*

Choudhuri, Sandipan, Kaustav Basu, Kevin Thomas, and Arunabha Sen. 2019. "Predicting Future Opioid Incidences Today." *arXiv preprint arXiv:1906.08891.*

Conitzer, Vincent, and Tuomas Sandholm. 2006. "Computing the optimal strategy to commit to." In *Proceedings of the 7th ACM conference on Electronic commerce.*

Cooper, Harry. 2016. "15,000 on French Terror Watchlist: Report." https://www.politico.eu/article/15000-on-french-terror-watchlist-report-radical-islamist/.

Deka, Deepjyoti, Ross Baldick, and Sriram Vishwanath. 2015. "Optimal data attacks on power grids: Leveraging detection & measurement jamming." In *2015 IEEE International Conference on Smart Grid Communications.* IEEE.

Deng, Ruilong, Gaoxi Xiao, Rongxing Lu, Hao Liang, and Athanasios V Vasilakos. 2016. "False data injection on state estimation in power systems—Attacks, impacts, and defense: A survey." *IEEE Transactions on Industrial Informatics* 13 (2): 411–423.

Dong, Ming, Bolong Zheng, Nguyen Quoc Viet Hung, Han Su, and Guohui Li. 2019. "Multiple Rumor Source Detection with Graph Convolutional Networks." In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management,* 569–578.

Eliades, Demetrios G, and Marios M Polycarpou. 2009. "A fault diagnosis and security framework for water systems." *IEEE Transactions on Control Systems Technology* 18 (6): 1254–1265.

Farajtabar, Mehrdad, Jiachen Yang, Xiaojing Ye, Huan Xu, Rakshit Trivedi, Elias Khalil, Shuang Li, Le Song, and Hongyuan Zha. 2017. "Fake news mitigation via point process based intervention." *arXiv preprint arXiv:1703.07823.*

Fooner, Michael. 1985. *A Guide to Interpol: The International Criminal Police Organization in the United States.* US Department of Justice, National Institute of Justice Washington, DC.

Foucaud, Florent. 2015. "Decision and approximation complexity for identifying codes and locating-dominating sets in restricted graph classes." *Journal of discrete algorithms* 31:48–68.

Foucaud, Florent, and Guillem Perarnau. 2011. "Bounds for identifying codes in terms of degree parameters." *arXiv preprint arXiv:1103.3756.*

Freeman, Linton C. 1978. "Centrality in social networks conceptual clarification." *Social networks* 1 (3): 215–239.

Frieze, Alan, Ryan Martin, Julien Moncel, Miklós Ruszinkó, and Cliff Smyth. 2007. "Codes identifying sets of vertices in random networks." *Discrete Mathematics* 307 (9-10): 1094–1107.

Fu, Julei, Ying Fan, Yang Wang, and Shouyang Wang. 2014. "Network analysis of terrorist activities." *Journal of Systems Science and Complexity* 27 (6): 1079–1094.

Gialampoukidis, Ilias, George Kalpakis, Theodora Tsikrika, Stefanos Vrochidis, and Ioannis Kompatsiaris. 2016. "Key player identification in terrorism-related social media networks using centrality measures." In *2016 European Intelligence and Security Informatics Conference (EISIC),* 112–115. IEEE.

Gravier, Sylvain, Ralf Klasing, and Julien Moncel. 2008. "Hardness results and approximation algorithms for identifying codes and locating-dominating codes in graphs." *Algorithmic Operations Research* 3 (1): 43–50.

Gupta, Aditi, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. "Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy." In *Proceedings of the 22nd international conference on World Wide Web,* 729–736. ACM.

Hancke, Gerhard P, Gerhard P Hancke Jr, et al. 2013. "The role of advanced sensing in smart cities." *Sensors* 13 (1): 393–425.

Hart, William E, and Regan Murray. 2010. "Review of sensor placement strategies for contamination warning systems in drinking water distribution systems." *Journal of Water Resources Planning and Management* 136 (6): 611–619.

Heber, Anita. 2009. "The networks of drug offenders." *Trends in Organized Crime* 12 (1): 1–20.

Hebrard, Emmanuel, Brahim Hnich, Barry O'Sullivan, and Toby Walsh. 2005. "Finding diverse and similar solutions in constraint programming." In *AAAI,* 5:372–377.

Hughes, Caitlin E, David A Bright, and Jenny Chalmers. 2017. "Social network analysis of Australian poly-drug trafficking networks: How do drug traffickers manage multiple illicit drugs?" *Social Networks* 51:135–147.

Interpol. n.d. "Organized Crime." https://www.interpol.int/en/Crimes/Organized-crime.

Jajodia, Sushil, Anup K Ghosh, Vipin Swarup, Cliff Wang, and X Sean Wang. 2011. *Moving target defense: creating asymmetric uncertainty for cyber threats.* Vol. 54. Springer Science & Business Media.

Jin, Zhiwei, Juan Cao, Han Guo, Yongdong Zhang, and Jiebo Luo. 2017. "Multimodal fusion with recurrent neural networks for rumor detection on microblogs." In *Proceedings of the 25th ACM international conference on Multimedia,* 795–816. ACM.

Jin, Zhiwei, Juan Cao, Yongdong Zhang, and Jiebo Luo. 2016. "News verification by exploiting conflicting social viewpoints in microblogs." In *Thirtieth AAAI Conference on Artificial Intelligence.*

Jones, Kevin D, Anamitra Pal, and James S Thorp. 2014. "Methodology for performing synchrophasor data conditioning and validation." *IEEE Transactions on Power Systems* 30 (3): 1121–1130.

Karnouskos, Stamatis. 2011. “Stuxnet worm impact on industrial cyber-physical system security.” In *Annual Conference of the IEEE Industrial Electronics Society.*

Karpovsky, Mark G, Krishnendu Chakrabarty, and Lev B Levitin. 1998. “On a new class of codes for identifying vertices in graphs.” *IEEE transactions on information theory* 44 (2): 599–611.

Kauppinen, Ari, Tarja Pitkänen, Haider Al-Hello, Leena Maunula, Anna-Maria Hokajärvi, Ruska Rimhanen-Finne, and Ilkka T Miettinen. 2019. “Two drinking water outbreaks caused by wastewater intrusion including sapovirus in Finland.” *International journal of environmental research and public health* 16 (22): 4376.

Kentucky, University of. 2001. “Kentucky Water Resources Research Institute.” https://uknowledge.uky.edu/kwrri/.

Khattar, Dhruv, Jaipal Singh Goud, Manish Gupta, and Vasudeva Varma. 2019. “MVAE: Multimodal Variational Autoencoder for Fake News Detection.” In *The World Wide Web Conference,* 2915–2921. ACM.

Khuller, Samir, Anna Moss, and Joseph Seffi Naor. 1999. “The budgeted maximum coverage problem.” *Information processing letters* 70 (1): 39–45.

Kim, Sukun, Shamim Pakzad, David Culler, James Demmel, Gregory Fenves, Steven Glaser, and Martin Turon. 2007. “Health monitoring of civil infrastructures using wireless sensor networks.” In *Proceedings of the 6th international conference on Information processing in sensor networks,* 254–263.

Kleinberg, Jon, and Eva Tardos. 2006. *Algorithm design.* Pearson Education India.

Kotschick, Dieter. 2006. “The topology and combinatorics of soccer balls: when mathematicians think about soccer balls, the number of possible designs quickly multiplies.” *American Scientist* 94 (4): 350–357.

Krause, Andreas, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos. 2008. “Efficient sensor placement optimization for securing large water distribution networks.” *Journal of Water Resources Planning and Management* 134 (6): 516–526.

Krebs, Valdis E. 2002. “Mapping networks of terrorist cells.” *Connections* 24 (3): 43–52.

Laifenfeld, Moshe, and Ari Trachtenberg. 2008. “Identifying codes and covering problems.” *IEEE Transactions on Information Theory* 54 (9): 3929–3950.

Laifenfeld, Moshe, Ari Trachtenberg, Reuven Cohen, and David Starobinski. 2009. "Joint monitoring and routing in wireless sensor networks using robust identifying codes." *Mobile Networks and Applications* 14 (4): 415–432.

Law, King. 2013. "How Drug Dealers Get Caught." https://www.robertkinglawfirm.com/blog/2013/november/how-drug-dealers-get-caught/.

Lee, Byoung Ho, and Rolf A Deininger. 1992. "Optimal locations of monitoring stations in water distribution system." *Journal of Environmental Engineering* 118 (1): 4–16.

Leskovec, Jure, and Andrej Krevl. 2014. *SNAP Datasets: Stanford Large Network Dataset Collection.* http://snap.stanford.edu/data, June.

Li, Bo, Dan Wang, Yiqing Ni, et al. 2009. "On the high quality sensor placement for structural health monitoring." In *Proc. IEEE INFOCOM,* 1–2. Citeseer.

Liang, Dieyan, Hong Shen, and Lin Chen. 2021. "Maximum Target Coverage Problem in Mobile Wireless Sensor Networks." *Sensors* 21 (1): 184.

Liebig, Jessica, and Asha Rao. 2014. "Identifying influential nodes in bipartite networks using the clustering coefficient." In *2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems,* 323–330. IEEE.

Lynch, Jerome P, and Kenneth J Loh. 2006. "A summary review of wireless sensors and sensor networks for structural health monitoring." *Shock and Vibration Digest* 38 (2): 91–130.

Mitchell, George J. 2007. "Report to the Commissioner of Baseball of an Independent Investigation Into the Illegal Use of Steroids and Other Performance Enhancing Substances by Players in Major League Baseball." http://mlb.mlb.com/mlb/news/mitchell/index.jsp.

Moncel, Julien. 2006. "On graphs on n vertices having an identifying code of cardinality $\lceil log_2(n+1)\rceil$." *Discrete Applied Mathematics* 154 (14): 2032–2039.

Müller, Tobias, and J-S Sereni. 2009. "Identifying and locating–dominating codes in (random) geometric networks." *Combinatorics, Probability and Computing* 18 (6): 925–952.

Murthy, Cherukuri, K Ajay Varma, Diptendu Sinha Roy, and Dusmanta Kumar Mohanta. 2014. "Reliability evaluation of phasor measurement unit using type-2 fuzzy set theory." *IEEE Systems Journal* 8 (4): 1302–1309.

Nandanoori, Sai Pushpak, Soumya Kundu, Seemita Pal, Khushbu Agarwal, and Sutanay Choudhury. 2020. "Model-Agnostic Algorithm for Real-Time Attack Identification in Power Grid using Koopman Modes." *Arxiv 2007.11717.*

Natarajan, Mangai. 2000. "Understanding the structure of a drug trafficking organization: a conversational analysis." *Crime Prevention Studies* 11:273–298.

———. 2006. "Understanding the structure of a large heroin distribution network: A quantitative analysis of qualitative data." *Journal of Quantitative Criminology* 22 (2): 171–192.

News, Fox. 2016. "Crews battle massive transformer fire at Avondale SRP substation." https://www.fox10phoenix.com/news/crews-battle-massive-transformer-fire-at-avondale-srp-substation.

Niu, Luyao, and Andrew Clark. 2019. "A Framework for Joint Attack Detection and Control Under False Data Injection." In *International Conference on Decision and Game Theory for Security,* 352–363. Springer.

Noel, Adam B., Abderrazak Abdaoui, Tarek Elfouly, Mohamed Hossam Ahmed, Ahmed Badawy, and Mohamed S. Shehata. 2017. "Structural Health Monitoring Using Wireless Sensor Networks: A Comprehensive Survey." *IEEE Communications Surveys Tutorials* 19 (3): 1403–1423. https://doi.org/10.1109/COMST.2017.2691551.

Ostfeld, Avi, James G Uber, Elad Salomons, Jonathan W Berry, William E Hart, Cindy A Phillips, Jean-Paul Watson, Gianluca Dorini, Philip Jonkergouw, Zoran Kapelan, et al. 2008. "The battle of the water sensor networks (BWSN): A design challenge for engineers and algorithms." *Journal of water resources planning and management* 134 (6): 556–568.

Padhee, Malhar, Joydeep Banerjee, Kaustav Basu, Sohini Roy, Anamitra Pal, and Arunabha Sen. 2018. "A new model to analyze power system dependencies." In *2018 IEEE Texas Power and Energy Conference (TPEC),* 1–6. IEEE.

Padhee, Malhar, Reetam Sen Biswas, Anamitra Pal, Kaustav Basu, and Arunabha Sen. 2020. "Identifying Unique Power System Signatures for Determining Vulnerability of Critical Power System Assets." *ACM SIGMETRICS Performance Evaluation Review* 47 (4): 8–11.

Pal, Anamitra, Chetan Mishra, Anil Kumar S Vullikanti, and SS Ravi. 2017. "General optimal substation coverage algorithm for phasor measurement unit placement in practical systems." *IET Generation, Transmission & Distribution* 11 (2): 347–353.

Pal, Anamitra, Anil Kumar S Vullikanti, and Sekharipuram S Ravi. 2016. "A PMU placement scheme considering realistic costs and modern trends in relaying." *IEEE Transactions on Power Systems* 32 (1): 552–561.

Paluch, Robert, Łukasz G Gajewski, Janusz A Hołyst, and Boleslaw K Szymanski. 2020. "Optimizing sensors placement in complex networks for localization of hidden signal source: A review." *Future Generation Computer Systems* 112:1070–1092.

Paruchuri, Praveen, Jonathan P Pearce, Janusz Marecki, Milind Tambe, Fernando Ordonez, and Sarit Kraus. 2008. "Playing games for security: An efficient exact algorithm for solving Bayesian Stackelberg games." In *AAMAS,* 2:895–902.

Potteiger, Bradley, Feiyang Cai, Abhishek Dubey, Xenofon Koutsoukos, and Zhenkai Zhang. 2020. "Security in Mixed Time and Event Triggered Cyber-Physical Systems using Moving Target Defense." In *IEEE International Symposium on Real-Time Distributed Computing.* IEEE.

Potthast, Martin, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2017. "A stylometric inquiry into hyperpartisan and fake news." *arXiv preprint arXiv:1702.05638.*

Qi, Peng, Juan Cao, Tianyun Yang, Junbo Guo, and Jintao Li. 2019. "Exploiting Multi-domain Visual Information for Fake News Detection." *arXiv preprint arXiv:1908.04472.*

Quadar, Nordine, Abdellah Chehri, Gwanggil Jeon, and Awais Ahmad. 2021. "Smart water distribution system based on IoT networks, a critical review." *Human Centred Intelligent Systems,* 293–303.

Racz, Miklos Z, and Jacob Richey. 2020. "Rumor source detection with multiple observations under adaptive diffusions." *IEEE Transactions on Network Science and Engineering.*

Ray, Saikat, David Starobinski, Ari Trachtenberg, and Rachanee Ungrangsi. 2004. "Robust location detection with sensor networks." *IEEE Journal on Selected Areas in Communications* 22 (6): 1016–1025.

Ray, Saikat, Rachanee Ungrangsi, De Pellegrini, Ari Trachtenberg, and David Starobinski. 2003. "Robust location detection in emergency sensor networks." In *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428),* 2:1044–1053. IEEE.

Ressler, Steve. 2006. "Social network analysis as an approach to combat terrorism: Past, present, and future research." *Homeland Security Affairs* 2 (2).

Rossi, Ryan A., and Nesreen K. Ahmed. 2016. "An Interactive Data Repository with Visual Analytics." *SIGKDD Explor.* 17 (2): 37–41. http://networkrepository.com.

Salehi, Vahid, Ahmed Mohamed, Ali Mazloomzadeh, and Osama A Mohammed. 2012. "Laboratory-based smart power system, part II: Control, monitoring, and protection." *IEEE Transactions on Smart Grid* 3 (3): 1405–1417.

Sato, Ryoma, Makoto Yamada, and Hisashi Kashima. 2019. "Approximation ratios of graph neural networks for combinatorial problems." *Advances in Neural Information Processing Systems* 32.

Sen, A, S Roy, K Basu, S Adeniye, S Choudhuri, and A Pal. 2021. "Optimal Cost Network Design for Bounded Delay Data Transfer from PMU to Control Center." In *2021 IEEE Global Communications Conference (GLOBECOM),* 1–6. IEEE.

Sen, Arunaba, Chenyang Zhou, Anisha Mazumder, Arun Das, Kaustav Basu, and Krzysztof Walkowiak. 2020. "On the Number of Steiner Trees in a Graph." In *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020,* 1–5. IEEE.

Sen, Arunabha, and Kaustav Basu. 2019. "On connectivity of interdependent networks." In *2019 IEEE Global Communications Conference (GLOBECOM),* 1–6. IEEE.

Sen, Arunabha, Sandipan Choudhuri, and Kaustav Basu. 2020. "Structural Dependency Aware Service Chain Mapping for Network Function Virtualization." In *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020,* 1–6. IEEE.

Sen, Arunabha, Victoria H Goliber, Kaustav Basu, Chenyang Zhou, and Sumitava Ghosh. 2019. "On upper and lower bounds of identifying code set for soccer ball graph with application to satellite deployment." In *Proceedings of the 20th International Conference on Distributed Computing and Networking,* 307–316. ACM.

Sen, Arunabha, Victoria Horan Goliber, Chenyang Zhou, and Kaustav Basu. 2018. "Terrorist Network Monitoring with Identifying Code." In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation,* 329–339. Springer.

Sengupta, Sailik, Kaustav Basu, Arunabha Sen, and Subbarao Kambhampati. 2020. "Moving target defense for robust monitoring of electric grid transformers in adversarial environments." In *International Conference on Decision and Game Theory for Security,* 241–253. Springer.

Sengupta, Sailik, Tathagata Chakraborti, and Subbarao Kambhampati. 2019. "MT-Deep: Moving Target Defense to Boost the Security of Deep Neural Nets Against Adversarial Attacks." *International Conference on Decision and Game Theory for Security.*

Sengupta, Sailik, Ankur Chowdhary, Dijiang Huang, and Subbarao Kambhampati. 2018. "Moving target defense for the placement of intrusion detection systems in the cloud." In *International Conference on Decision and Game Theory for Security,* 326–345. Springer.

Sengupta, Sailik, Ankur Chowdhary, Abdulhakim Sabur, Adel Alshamrani, Dijiang Huang, and Subbarao Kambhampati. 2020. "A survey of moving target defenses for network security." *IEEE Communications Surveys & Tutorials.*

Sengupta, Sailik, Satya Gautam Vadlamudi, Subbarao Kambhampati, Adam Doupé, Ziming Zhao, Marthony Taguinod, and Gail-Joon Ahn. 2017. "A Game Theoretic Approach to Strategy Generation for Moving Target Defense in Web Applications." In *AAMAS,* 178–186.

Shah, Devavrat, and Tauhid Zaman. 2011. "Rumors in a network: Who's the culprit?" *IEEE Trans. on information theory* 57 (8): 5163–5181.

Shi, Baoxu, and Tim Weninger. 2016. "Fact checking in heterogeneous information networks." In *Proceedings of the 25th International Conference Companion on World Wide Web,* 101–102. International World Wide Web Conferences Steering Committee.

Shu, Kai, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. "Fake news detection on social media: A data mining perspective." *ACM SIGKDD Explorations Newsletter* 19 (1): 22–36.

Shu, Kai, Suhang Wang, and Huan Liu. 2018. "Understanding user profiles on social media for fake news detection." In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR),* 430–435. IEEE.

Sinha, Arunesh, Thanh H Nguyen, Debarun Kar, Matthew Brown, Milind Tambe, and Albert Xin Jiang. 2015. "From physical security to cybersecurity." *Journal of Cybersecurity* 1 (1): 19–35.

Spinelli, Brunella, L Elisa Celis, and Patrick Thiran. 2017. "A general framework for sensor placement in source localization." *IEEE Transactions on Network Science and Engineering* 6 (2): 86–102.

Suomela, Jukka. 2007. "Approximability of identifying codes and locating–dominating codes." *Information Processing Letters* 103 (1): 28–33.

Tacchini, Eugenio, Gabriele Ballarin, Marco L Della Vedova, Stefano Moret, and Luca de Alfaro. 2017. "Some like it hoax: Automated fake news detection in social networks." *arXiv preprint arXiv:1704.07506.*

Tan, Song, Debraj De, Wen-Zhan Song, Junjie Yang, and Sajal K Das. 2017. "Survey of security advances in smart grid: A data driven approach." *IEEE Communications S&T.*

Tang, Wenchang. 2020. "Identifying misinformation and their sources in social networks."

Team, Symantec. 2017. *Dragonfly: Western energy sector targeted by sophisticated attack group.*

Tripathi, Abhishek, Hari Prabhat Gupta, Tanima Dutta, Rahul Mishra, KK Shukla, and Satyabrat Jit. 2018. "Coverage and connectivity in WSNs: A survey, research issues and challenges." *IEEE Access* 6:26971–26992.

UCINET. 2022. "Covert Networks." https://sites.google.com/site/ucinetsoftware/datasets/covert-networks.

UNODC. 2018. "UNODC World Report." https://www.unodc.org/wdr2018/en/exsum.html.

———. n.d. "Drug Trafficking." https://www.unodc.org/unodc/en/drug-trafficking/.

Vazirani, Vijay V. 2013. *Approximation algorithms.* Springer Science & Business Media.

Wang, Bang. 2011. "Coverage problems in sensor networks: A survey." *ACM Computing Surveys (CSUR)* 43 (4): 1–53.

Wang, Yaqing, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. "Eann: Event adversarial neural networks for multi-modal fake news detection." In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining,* 849–857. ACM.

Wang, Zheng, Chaokun Wang, Jisheng Pei, and Xiaojun Ye. 2017. "Multiple source detection without knowing the underlying propagation model." In *Thirty-First AAAI Conference on Artificial Intelligence.*

Wikipedia. 2022. "List of Terrorist Incidents in France." https://en.wikipedia.org/wiki/List_of_terrorist_incidents_in_France.

———. n.d. "Metcalf Sniper Attack." https://en.wikipedia.org/wiki/Metcalf_sniper_attack.

Xiao, Ying, Christoforos Hadjicostis, and Krishnaiyan Thulasiraman. 2006. "The d-identifying codes problem for vertex identification in graphs: probabilistic analysis and an approximation algorithm." In *International Computing and Combinatorics Conference,* 284–298. Springer.

Yang, Shuo, Kai Shu, Suhang Wang, Renjie Gu, Fan Wu, and Huan Liu. 2019. "Unsupervised fake news detection on social media: A generative approach." In *Proceedings of 33rd AAAI Conference on Artificial Intelligence.*

Zhou, Chenyang, Anisha Mazumder, Arun Das, Kaustav Basu, Navid Matin-Moghaddam, Saharnaz Mehrani, and Arunabha Sen. 2018. "Relay node placement under budget constraint." In *Proceedings of the 19th International Conference on Distributed Computing and Networking,* 1–11.

Zhou, Yousheng, Chujun Wu, Qingyi Zhu, Yong Xiang, and Seng W Loke. 2019. "Rumor source detection in networks based on the SEIR model." *IEEE access* 7:45240–45258.

Zhu, Kai, Zhen Chen, and Lei Ying. 2016. "Catch'em all: Locating multiple diffusion sources in networks with partial observations." *arXiv preprint arXiv:1611.06963.*

Zhu, Kai, and Lei Ying. 2014. "Information source detection in the SIR model: A sample-path-based approach." *IEEE/ACM Transactions on Networking* 24 (1): 408–421.

Zimmerman, Ray Daniel, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas. 2010. "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education." *IEEE Transactions on power systems* 26 (1): 12–19.

APPENDIX A

OTHER RESEARCH COLLABORATIONS

Apart from studying problems concerning monitoring of complex networks for anomalous behavior, I was fortunate enough to collaborate with other researchers on multiple problem statements. Here, in this appendix, I provide an overview of my other contributions.

A.1 Interdicting Illicit Networks

The objective here was to design approaches in order apprehend traffickers trafficking victims from other parts of United States to Las Vegas. Due to the lack of real world data, we had to design approaches to determine the paths taken by the traffickers between a source-destination pair, which would result in the least probability of interdiction. Next, once such paths were computed for multiple source-destination pairs, our approach was to determine the maximum payoffs for the law enforcement agents, who were limited by an operational budget. Parameters such as, law enforcement budget, traffickers budget (the trafficker might not want to spend more than $X traveling between two cities), probabilities of interdiction, etc. were approximated due to the lack of real world data.

A.2 Predict Future Opioid Overdose Incidences

In collaboration with the Arizona Health Care Cost Containment System (AHCCCS), we studied the effect of the Opioid epidemic in the state of Arizona. Opioids are drugs given as pain relievers and it has been known to cause addiction in individuals. We observed how the number of Opioid overdoses in the state kept increasing over the past decade. The goal of this collaboration was to develop machine learning based predictive tools in order to accurately predict future instances of Opioid overdoses, such that, state and local government can allocate appropriate resources. We developed synthetic datasets, with Arizona characteristics (after discussions with third party agencies), as real world data was difficult to procure due to various privacy constraints. We not only showed how machine learning can be utilized for this task but also showed that Twitter data can be used as a real time indicator of overdosing incidences in the state. This is because, we observed a high correlation between the number of tweets and Opioid overdosing and Opioid related deaths in the state.

A.3 Communication Network Design in Power Grids

Sensors, such as the Phasor Measurement Units (PMUs), deployed in the power grid, must send data and communicate with local control centers, in order for the power grid operator to monitor the operational behavior of the grid. This communication is bounded by a delay value δ, which implies that the data must be received at the control center within δ time units. This problem has been modeled as Rooted Delay Constrained Spanning Tree problem and we have provided an optimal as well as a heuristic, which performs almost as well as the optimal solution. Current work in this domain include the scenario where there are multiple control centers present (instead of one) as well as viewing the design of the communication network and placing the sensors in the power grid as a multi-objective problem.