INTEGER OPTIMIZATION FOR THE UNDERSTANDING AND DISRUPTION OF ILLICIT NETWORKS

Daniel Kosmas

Submitted in Partial Fullfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

Approved by:
John E. Mitchell, Chair
Kristin P. Bennett
Yangyang Xu
Thomas C. Sharkey

Department of Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, New York

[August 2022]
Submitted June 2022

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to first thank my doctoral committee, Dr. John Mitchell, Dr. Kristin Bennett, Dr. Yangyang Xu, and Dr. Thomas Sharkey. I would like to extend special thanks to my advisors, Dr. John Mitchell and Dr. Thomas Sharkey, for the time and energy they have put in to supporting me and helping me grow as a researcher. They are both better advisors than I could have hoped for, and I am grateful for the opportunity to work with both of them. I would also like to thank the all the educators in my life that have supported me in my academic career.

It is important that we recognize that any research cannot capture all of the complexities of the lived experiences of trafficking victims and survivors, and only with the insights of subject matter experts can be more accurately model these complexities. I would like to thank our survivor-centered advisory group for their significant contributions in supporting my research, including: Terry Forliti, Breaking Free, CEO/Survivor Leader; Tonique Ayler, Breaking Free, Housing Advocate/Survivor Leader; Joy Friedman, Independent Consultant, Survivor Leader; Christine Nelson, Independent Consultant, Fellow of Survivor Alliance/Survivor Leader; Drea Sortillon, Hennepin County Attorney's Office, Witness-Victim Division; Lorena Pinto, Vice President of Programs, The Family Partnership; and Mikki Mariotti, Director of the PRIDE Program of the Family Partnership. I would also like to thank the National Institute of Justice (NIJ) for supporting my research with the NIJ Graduate Research Fellowship.

I am especially grateful for the support of my parents. From raising me to put academics first to supporting me through college and graduate school, I would have never been able to be the person I am or scholar I am without them. I would also like to thank my friends for their support, from late nights staying up working on homework and projects to board game nights and Dungeons and Dragons adventures.

ABSTRACT

Tools from operations research (OR) have historically been applied to better understand and disrupt illicit trafficking networks. Two such tools are community detection and network interdiction. Community detection has been used to aid in understanding the roles of participants of illicit networks. However, edges may be hidden in an attempt to prevent correct classifications of participants in the network. Network interdiction models have been applied to disrupting illicit networks, such as drug smuggling and nuclear smuggling networks. Current network interdiction models do not allow for the networks to change after disruption decisions have been implemented, which is contrary to how we expect these networks to behave. This is especially important since we will be applying network interdiction models to disrupting human trafficking networks. There are many opportunities for the OR community to help address human trafficking. However, there are many challenges associated with applying OR tools to these networks. One of the major challenges associated with applying operations research models to disrupting human trafficking networks is a limited amount of reliable data sources readily available for public use, since operations are intentionally hidden to prevent detection, and data from known operations are often incomplete.

This dissertation considers new integer optimization problems to better understand and disrupt illicit trafficking networks. We first consider two new problems regarding how edge addition or removal impacts the modularity of partitions (or community structures) in a network. The first problem seeks to add edges to enforce that desired partitions are the the partitions with maximum modularity. The second problem seeks to find the sparsest representation of a network that has the same partition with maximum modularity as the original network. We present integer programming formulations of these problems, as well as heuristic algorithms to solve them. We then consider a new class of max flow network interdiction problems, where the defender is able to introduce new arcs to the network after the attacker has made their interdiction decisions. We provide an example of when interdiction can result in an increase to the maximum flow, and prove properties of when this restructuring will not increase the value of the minimum cut, which is known to be equivalent to the maximum flow. This has important practical interpretations for problems of disrupting drug or human trafficking networks. In particular, it demonstrates that disrupting lower levels of

these networks will not impact their operations when replacing the disrupted participants is easy. For the bilevel mixed integer linear programming formulation of this problem, we devise a column-and-constraint generation (C&CG) algorithm to solve it. Our approach uses *partial information* on the feasibility of restructuring plans and is shown to be orders of magnitude faster than previous C&CG methods. We then extend this problem to include a temporal component, where interdictions and restructurings are implemented throughout the time horizon. Modeling adaptations are proposed to reduce the size of the problem, and modeling-based augmentations are implemented in the proposed C&CG algorithm to further improve the solve time of our method. To apply our models to the disruption of sex trafficking networks, we propose a novel conceptualization of flow as the ability of a trafficker to control their victims. Our results show the importance of understanding how sex traffickers react to disruptions, especially in terms of recruiting new victims.

Additionally, we help address the data problem in human trafficking by proposing a network generator for domestic sex trafficking networks by integrating OR concepts and qualitative research. Multiple sources have been triangulated to ensure that networks produced by the generator are realistic, including law enforcement case file analysis, interviews with domain experts, and a survivor-centered advisory group with first-hand knowledge of sex trafficking. The output models the relationships between traffickers, so-called "bottoms", and victims. This generator allows operations researchers to access realistic sex trafficking network structures in a responsible manner that does not disclose identifiable details of the people involved. We demonstrate how the networks produced by the network generator can be used as data for the network interdiction models.

CHAPTER 1
INTRODUCTION

The existence of illicit trafficking networks has long been a challenge facing society due to the illicit activities they support, such as drug trafficking and human trafficking, and the violence that sustains their operations [60]. For example, illicit drugs and the trafficking networks that distribute them pose a significant threat to law enforcement and public health. According to the National Drug Threat Analysis [44], drug poisoning deaths are the leading cause of injury death in the United States, and drug sales account for tens of billions of dollars annually in illicit profits. These organizations can also further perpetuate violence through the funding of terrorist organizations [28]. Numerous analytical tools have been applied to better understand how illicit networks, such as drug smuggling networks, function and how relevant stakeholders can best allocate resources to disrupt these networks. Bahulkar et. al. [6] proposes an integrative framework for the detection and disruption of transnational illicit trafficking organizations that brings together social network analysis and optimization. In this dissertation, we expand upon two of these tools, community detection and network interdiction models.

Law enforcement agencies have been turning to social network analysis in order to augment their investigations [5, 31, 38, 87, 114, 116, 120]. In particular, the problem of edge augmentation is of high importance to law enforcement; it is known that participants in illicit networks attempt to hide their activities from law enforcement in order to avoid detection [11, 52, 113]. However, missing edges can dramatically change the output of these quantitative tools, making the problem of identifying missing edges fundamentally important for the appropriate use of social network analysis in law enforcement investigations [120]. The problem of identifying missing edges in illicit networks has been approached from many different perspectives [12, 23, 82, 103].

Portions of this chapter previously appeared as: D. Kosmas, J.E. Mitchell, T.C. Sharkey, and B.K. Szymanski, *Optimizing edge sets in networks to produce ground truth communities based on modularity*, Networks, (2021) pp. 1-26. https://doi.org/10.1002/net.22084. ©2021 John Wiley & Sons, Inc.,
D. Kosmas, T.C. Sharkey, J.E. Mitchell, K.L. Maass, and L. Martin, *Interdicting Restructuring Networks with Applications in Illicit Trafficking*, preprint, arXiv:2011.07093 [math.OC], 2020. Submitted to Eur. J. Oper. Res.,
D. Kosmas, C. Melander, E. Singerhouse, T.C. Sharkey, K.L. Maass, K. Barrick, and L. Martin, *Generating Synthetic but Realistic Human Trafficking Networks for Modeling Disruptions through Transdisciplinary and Community-Based Action Research*, preprint, arXiv:2203.01893 [cs.SI], 2022. Submitted to IISE Trans.

Another facet of using social network analysis in augmenting the investigations of illicit networks is through analyzing participant attributes. It has been identified that attributes of participants in illicit networks can provide direction for law enforcement investigations to understand the structure of these networks [21, 76, 91, 92]. In social network analysis, one way of uncovering potential node attributes is through community detection. The community detection problem seeks to partition the nodes into clusters, or communities, based on some measure [117]. Calderoni et al. [22] identified that attributes, such as the roles of participants, are linked to the communities identified through modularity maximization [96]. This connection has sparked further research into the application of modularity maximization to understanding illicit networks, based around the notion that incomplete information about networks impacts the communities identified [7, 46].

We introduce a new problem bridging the areas of edge augmentation and community detection. As law enforcement investigates an illicit network, they may not identify all of the edges [11, 113]. This, in turn, hinders the performance of community detection methods [7], potentially causing the communities identified to disagree with the roles determined by the investigations. We seek to identify the minimum number of edges to add to the network to enforce that the roles determined by law enforcement investigations align with the communities determined via community detection. From here on, we refer to known communities from external investigation as the *ground truth community structure*, or *ground truth partition*.

We also consider a similar problem when law enforcement has observed parts of the illicit network, but is not yet confident about the roles of the participants. In this case, social network analysis is used to predict the roles of participants in an illicit network [22], and law enforcement must then direct its investigations to validate these predictions. However, law enforcement is limited in its resources to perform all its duties [41, 55, 61]. These resource constraints impact the quality of investigations [37, 53]. In evaluating the quality of the communities determined via social network analysis, it is important that law enforcement validates these findings as efficiently as possible. This problem can be framed as identifying a sparse sub-network which maintains the partition of the original network. In other words, we seek to find the smallest number of edges (amongst the current ones) such that the partition that maximizes the modularity of the sub-network is the same as the partition that maximizes modularity of the original network. Understanding the minimum number

of edges needed to produce these communities will allow scarce resources to be allocated to best investigate the edges of the network to prove or disprove what is believed by the investigations.

Network interdiction models have been applied to the disruption of nuclear smuggling [90] and drug trafficking networks. Malaviya et. al. [75] first applied network interdiction models to the disruption of city-level drug trafficking networks and described a framework for generating networks representative of city-level drug trafficking networks, where we interdict nodes, instead of arcs. Baycik et. al. [9] expanded upon this work to include the disruption of drug trafficking networks where not all participants in the network directly handle the drugs. In this work, an additional information network models how bosses direct the flow of drugs in the drug trafficking network. Shen et. al. [109] further expands upon the model by also including a money network to help understand the role of money laundering in drug trafficking networks. Jabarzare et al. [58] consider a new formulation for multi-period network interdiction for illicit supply chains, and apply a decomposition method to solve their formulation. In these works, the drug trafficking network does not have the ability to react to the disruptions. However, we can reasonably expect that the members of the drug trafficking network will have some sort of reaction, from drug dealers finding new users to sell their drugs to, to the operations of the suppliers being moved to backup locations. To be able to properly disrupt drug trafficking networks, we must make disruption decisions that incorporate likely reactions and actions the networks may make. We will demonstrate in our computational analysis the importance of accounting for the reaction of the drug trafficking network.

The dynamics of the network are important to consider in disrupting illicit trafficking networks [20]. We examine the max flow network interdiction problem (MFNIP) in relation to drug trafficking networks and propose a way to apply MFNIP to the victim recruitment in human trafficking for sexual exploitation (herein referred to as "sex trafficking"). Drug trafficking results in billions of dollars in illicit profits, constituting a fifth of all revenue from organized crime [28]. It has been observed that when law enforcement targets low-level drug offenders, such as dealers or users, there is little impact on the amount of drugs sold. This is because new participants can be easily recruited to replace those who were incarcerated [14, 16]. Current network interdiction models for disrupting drug trafficking networks do not account for the ability to replace nodes and arcs. Magliocca et. al. [74] developed an

agent-based model to predict the response of drug traffickers to interdiction efforts, which is tailored to use by an individual agency. Our model uses the full operations of the drug trafficking network to plan global disruptions. This is particularly important in the case of sex trafficking, where "victim-level" interdictions, focusing resources on only interdicting victims, may result in an increase in the total number of people who have been trafficked due to not accounting for trafficker recruiting new victims [27].

The remainder of this chapter is organized as follows. Section 1.1 discusses sex trafficking and some of the challenges of applying tools from operations research to sex trafficking. Section 1.2 introduces community detection through modularity maximization. Section 1.3 introduced max flow network interdiction. Section 1.4 outlines the remainder of this dissertation. We note that, when discussing modularity maximization, we refer to networks consisting of nodes and edges, and when discussing network interdiction, we refer to networks consisting of nodes and arcs. We do so to be consistent with the relevant literature, as modularity maximization is typically performed on undirected networks, whereas max flow network interdiction is typically modeled on directed networks.

1.1 Sex Trafficking and Operations Research

An emerging application of network interdiction is disrupting human trafficking networks. Human trafficking is human rights abuse and documented both in the United States and abroad. Although the magnitude of the problem is difficult to determine [47, 48], estimates suggest that nearly 40 million people were victims of human trafficking in 2016 [40]. Human trafficking involves the use of force, fraud, or coercion in order to exploit a person for the purposes of labor or services (labor trafficking) or sexual exploitation (sex trafficking) [26, 79, 4, 101]. Despite the growing awareness of the problem, there has been limited research into developing quantitative tools to address the problem of human trafficking [27, 43, 66]. This is partially due to a lack of readily-usable data to appropriately populate these tools. Since these networks are illicit in nature, their operations are hidden to avoid detection [47, 48, 65, 66]. Current data for analytic approaches often comes from case files, scraped web data on sex advertisements, or data pertaining to massage parlors [62, 83, 123]. This data can be viewed as the "forward-facing" aspects of the operations of sex trafficking networks in terms of how they interact with the outside world, thus it is (somewhat) publicly accessible. However, there are data gaps around the more internal-facing

operations and social connections between traffickers and victims.

There are some data sources that can be investigated regarding domestic sex trafficking networks, including case file analysis and interviews with those who have been a part of, or victimized by, these networks [24, 45, 69, 126]. However, no single source will have the full information necessary to perform a proper analysis of the network structure. For example, data collected by law enforcement is known to be incomplete. This is because the focus of law enforcement is on proving the occurrence of a specific crime and, thus, may ignore details that do not help this goal, but would be relevant to construct the full extent of the trafficking network [33]. Additionally, extracting the relevant information from the case files to produce the network takes a significant amount of time. Traffickers and survivors of sex trafficking are valuable sources of information on sex trafficking networks, but there are numerous challenges associated with gathering data from them. These include ethical and logistical challenges, as well as issues regarding the generalizability of the data [51, 122].

OR models for societal challenges can be supplemented with insights from domain experts from a wide array of disciplines, such as qualitative researchers. Transdisciplinary research is necessary to conduct appropriate research related to human trafficking networks and seeks to address complex societal challenges through the integration of knowledge and methods of different disciplines [70]. Because of the complexities of the lived experiences of survivors of sex trafficking, domain expertise is necessary to ensure that any analytical tools developed for the purpose of understanding and disrupting sex trafficking networks appropriately consider the human element of these networks. There are many challenges with building a transdisciplinary research team, but the time and effort put in to build such a team can result in scholarly works better grounded in the application area [108]. We have applied a transdisciplinary approach in creating the proposed network generator by collaborating with domain experts, who have been investigating human trafficking for over 10 years (see [79, 80, 81]), and a survivor-centered advisory group.

Incorporating domain expertise in the application of OR is critical since it allows the created models to focus on the true underlying problems faced in the application area. This is especially important when the system cannot be directly observed. Morris [88] discusses the types of tasks necessary to inform models while Willemain [124, 125] discuss the process by which experts create models when faced with a practical problem. However, less research has been done on how to integrate domain expertise, both within other academic disciplines and

from practitioners, on socially-sensitive, hidden issues, like human trafficking. Caulkins et al. [27] discuss the potential types of insights that can be obtained by using engineering models to understand human trafficking and discuss the need to partner with human trafficking domain experts. Sharkey et al. [108] present a high-level approach for how to integrate this expertise into the modeling process. A critical observation of Sharkey et al. [108] is that data is one of the key areas where experts need to inform the modeling process.

Dimas et al. [43] provides a comprehensive review of existing literature in applying operations research and analytics to the understanding and disruption of human trafficking. Researchers are currently considering investigating two different perspectives for sex trafficking networks. One perspective considers how sex trafficking networks interact with different locations. Mayorga et al. [83] used web scraping from an online escort service to model the movement of sex trafficking networks between different cities. Keskin et al. [62] proposed a process of grouping online advertisements to predict the movement of sex trafficking networks based on where related advertisements appear. White et al. [123] used web scraping to gather data on illicit massage parlors, and determines potential contributing factors for where illicit massage parlors may be located. These works use a visible, forward-facing part of the sex trafficking networks, the advertisements, to perform their analysis. Yet, advertisements are not the same as number of victims or people in the network and it is unclear exactly how advertisements map to the underlying phenomena of trafficking itself. The other perspective considers the relationships of the individuals within the sex trafficking network. Cockbain [32] performs social network analysis of traffickers and victims in six sex trafficking operations in the UK that law enforcement disrupted. Mancuso [77] investigated how a person's role in the sex trafficking network impacted their centrality measures by analyzing a case file of a sex trafficking network that spanned Italy to Nigeria. Campana [24] further explored the structure of a sex trafficking network, focusing on when multiple members of the network participated in the same trafficking events, by analyzing a case file on an international sex trafficking network that spanned Europe and Nigeria. We seek to augment the social network perspective by incorporating the insights of domain experts and people with lived experiences in sex trafficking networks, elucidating hidden aspects of domestic sex trafficking networks that may not be able to be determined by advertisements or law enforcement investigations.

1.2 Introduction to Modularity Maximization

Community detection has been a problem of interest in network analysis for many years, with many different methods [117]. Formally, consider a network $G = (V, E)$. A feasible partition of V is defined to be a set of clusters, or communities, $P = (C_1, \ldots, C_k)$ such that $\bigcup_{i=1}^{k} C_i = V$ and $C_i \cap C_j = \emptyset$ for all $i \neq j$. Let $\mathcal{P}$ be the set of feasible partitions. The community detection problem seeks to find a partition $P^* \in \mathcal{P}$ that maximizes some measure of the partition. Typically, this measure would give preference to partitions where there are a large number of edges within each cluster, but a small number of edges between the clusters. Outside of illicit network analysis, community detection has applications in many domains, including the analysis of biological networks and social networks [117].

A popular method to perform community detection is through modularity maximization [29], introduced by Newman and Girvan [96]. Modularity maximization seeks to find a partition P that maximizes the Newman-Girvan modularity metric, which compares, for each cluster, its number of actual edges and expected numbers of edges in the equivalent random network without communities [93]. Modularity was initially defined on unweighted networks, but the definition can easily be extended to weighted networks. Brandes et al. [19] provides an equivalent definition of the unweighted version of modularity as

$$Q = \sum_{C \in P} \left(\frac{|E(C)|}{|E|} - \left(\frac{|E(C)| + \sum_{C' \in P} |E(C, C')|}{2|E|} \right)^2 \right), \tag{1.1}$$

where $E(C)$ is the set of edges within cluster C and $E(C, C')$ is the set of edges between clusters C and C'. The problem of modularity maximization is known to be NP-hard [19]. Many heuristic methods have been developed to quickly find partitions with relatively large modularity scores [94, 15]. Similar metrics have also been proposed to capture the same intuitive principles guiding modularity, while accounting for some of its shortcomings [30].

Modularity maximization can be formulated as an integer programming (IP) problem. Brandes et al. [19] introduces this formulation, and Agarwal and Kempe [1] propose a rounding technique to improve the solve time of these problems. Given a weighted network $G = (V, E)$ with edge weights $w : E \to \mathbb{R}$, let A be the weighted adjacency matrix of G, $m = \sum_{(i,j) \in E} w_{ij}$, and $d_i = \sum_{j \in V} w_{ij}$ for $i \in V$. For every pair of nodes $i, j \in V$, define decision variables x_{ij} such that $x_{ij} = 0$ if i and j are in the same cluster, and $x_{ij} = 1$ otherwise. The integer programming formulation of modularity maximization is as follows:

$$\max_{x\in\{0,1\}^{|V|^2}} \frac{1}{2m}\sum_{i,j\in V}\left(A_{ij}-\frac{d_i d_j}{2m}\right)(1-x_{ij}) \tag{1.2a}$$

$$\text{s.t. } x_{ij}\le x_{ik}+x_{kj} \text{ for all } i,j,k\in V \tag{1.2b}$$

There have since been other formulations of this problem to improve solve time of the integer program (see [3] for such methods), but discussing them is outside the scope of this dissertation. Such methods can easily be adapted to fit into our framework.

One issue that arises in the modularity maximization problems considered in this dissertation, compared to this past work on using IP to maximize modularity, is the fact that we are enforcing that the modularity of the given partition is maximum amongst all partitions. This leads to a number of constraints equal to the number of partitions, i.e., an exponential number of constraints based on the number of nodes in the network. To avoid this difficulty, our framework operates on 'row generation,' i.e., only maintaining a subset of these constraints at a time and then generating new constraints (rows) when needed. In each iteration of our framework, a master problem is solved for a potential optimal solution based on a subset of the partition constraints. A subproblem is then solved to verify optimality, which fails when there is a partition not currently included that has a modularity larger than the ground truth community structure. In such a case, a new constraint is added to the master problem. Fischetti et al. [49] demonstrates this is an effective method to solve bilevel integer optimization problems. More recently, community detection through the degree-corrected stochastic block model, which is known to be equivalent to modularity maximization [95], has been modeled as a MINLP in [107]. The authors linearized the model and solved the resulting MILP using a row generation framework. However, our problem has the added complication of needing to compare the ground truth to all potential partitions in a network where the problem is deciding the existence of edges.

1.3 Introduction to Network Interdiction

Network interdiction is a two player Stackelberg game [115] between an interdictor (attacker) and an operator (defender). For the maximum flow network interdiction problem (MFNIP), the defender wishes to maximize the flow through the network, while the attacker attempts to minimize the flow by disrupting nodes and arcs (note that, without loss of gen-

erality, we can focus on the disruption of nodes in the network), removing them from the network [127]. Max flow network interdiction problems model situations where we want to move goods or services through a network, such as telecommunications, electrical power, transportation, and illicit trafficking networks [110]. Illicit trafficking networks, such as drug trafficking and human trafficking networks, are networks where the product or service being supplied is illegal. In illicit trafficking networks, the interdiction decisions would represent disruptions to the network, with the simplest example being law enforcement arresting participants within the network. It is important to recognize that not all participants in an illicit trafficking network, such as a drug trafficking network, are engaged in illicit activity, and that non-law enforcement based interventions, such as investment in communities and rehabilitation, can also be modeled as interdiction decisions. Therefore, for the purposes of this dissertation, the term 'interdiction' will be defined to be any disruptive action that alters the nodes and arcs within the illicit trafficking network.

An important class of network interdiction models, known as defender-attacker-defender models, allows for the defender to protect part of their network before the attacker makes interdiction decisions [2, 71]. However, these models do not allow for the network to change after the interdiction decisions have been made. Extending these models over a longer time period, once the network has been disrupted, the damage is permanent, and no attempts to repair the network are allowed [75]. In some applications, such as with illicit trafficking networks, this would contradict our intuition; if an illicit trafficking network has lost participants, and thus lost business, we would expect the network to restructure to compensate for those losses. Incorporating the restructuring decisions of the illicit trafficking network is important to providing appropriate recommendations on how law enforcement and other stakeholders should disrupt the illicit trafficking network. However, fully incorporating the dynamic back-and-forth between attackers and defenders is computationally difficult [102, 106]. In this dissertation, we offer an approach to capture more of the dynamics than traditional MFNIPs which helps to yield important practical insights into the problems of disrupting drug trafficking networks. We consider a variant of MFNIP where the defender is allowed to restructure parts of the network before operating it.

There has been limited work in applying network interdiction to human trafficking networks. Mayorga et al. [83] discusses a robust max flow network interdiction model, where there is uncertainty about the capacities of arcs. Tezcan and Maass [119] consider a multi-

period max flow network interdiction model where there is uncertainty about the success of the proposed interdictions, and the success of interdictions in each time period is dependent on the success of interdictions in the previous time periods. The model is unique in that they consider flow to be the desirability of a trafficker to travel along different routes. These models both model the movement of people across geographic areas. Xie and Aros-Vera [128] propose a network interdiction model on interdependent networks, similar to Baycik et al. [9]. Their model considers flow to be the victims in the network, being transported between people with different roles in the sex trafficking network. These models all focus on larger scale operations, and all fail to consider the autonomy of the victims themselves in the trafficking networks. The conceptualization we propose treats victims as more than a product to be moved through the network.

Uncertainty in max flow network interdiction has been a topic of interest for many years [35, 59]. More recently, Lei et al. [68] explored a variant of stochastic MFNIP where the defender is able to add capacity to arcs in the network after the uncertainty around capacities is resolved. They allow for continuous capacity to be added, which does not introduce the challenges of integer decisions in the lower level problem. Borrero et al. [17, 18] explore uncertainty in network structure. Ketkov and Prokopyev [63] additionally explore sequential interdiction in uncertain network structures and further consider how the optimal interdiction strategies change based on what type of player the defender is. These model does not allow for the underlying network to change, which is the adaptation we introduce and critical for our motivating applications.

Including a temporal component into network interdiction models has also been a topic of interest. Rad and Kakhki [102] propose a multi-period max flow network interdiction model where each arc has a traversal time for flow to travel across it, and derive a Benders' decomposition algorithm based on Temporally Repeated Flows to solve this problem. Soleimani-Alyar and Ghaffari-Hadigheh [111] consider a multi-period interdiction model where flow is sent from source to sink instantaneously, and solve the problem with generalized Benders' decomposition. They also expand this model to include uncertainty in the arc capacities [112]. Malaviya et al. [75] and Jabarzare [58] apply multi-period max flow network interdiction models to illicit drug trafficking networks, and Tezcan and Maass [119] and Xie and Aros-Vera [128] apply multi-period max flow network interdiction models to human trafficking networks.

Interdiction problems considered in this dissertation are similar to shortest path interdiction with improvement (SPIP-I) introduced by Holzmann and Smith [56], where arcs can be improved after interdiction decisions have been made. In their work, they construct a layered network to solve their problem as a shortest path interdiction problem. Fischetti et al. [49] further explores interdiction games, where the decisions of the interdictor explicitly restrict the actions of the operator under the assumption of downward monotonicity. In this work, they are able to apply a direct penalization to the objective function value of previous sub-problem solutions based on the current interdiction plan due to the interdiction decisions directly preventing the defender from taking certain actions. In our problems, the interdiction decisions do not restrict the actions of the defender, but instead allow for certain arcs to be added to the network. Further, generating constraints associated only with the objective function value based on each restructuring would not be correct, as the minimum cut may not be the same across different restructuring plans. This prevents their method from being applied to our problems. The interdiction problems studied in Contardo and Sefair [34] similarly have that the interdiction decisions directly penalize the actions of the defender, and likewise their solution method cannot be applied to our problems.

Our proposed network interdiction problems fall into the general class of bilevel integer programming problems. In general, bilevel integer programs are $\mathcal{NP}$-Hard problems, and are more difficult to solve than standard optimization problems because of the interactions between the two decision makers. When decisions are restricted to be integers, these problems become harder, because standard methods to solve non-integer counterparts cannot be used. Solution methods to general BIPs have been recently explored [50, 72, 118]. BIPs have been solved via C&CG methods, which separate the inner optimization into two problems, a 'middle' problem with just integer decision variables and one with continuous variables. By enumerating over the feasible integer decisions, we can condense the remaining two optimization problems into one by exploiting duality of linear programs. We then solve the optimization problem with a subset of these feasible integer decisions, and iteratively identify new integer decisions to add to the problem. Smith and Song [110] identify C&CG as a new method to solve network interdiction problems.

Zeng and An [131] first applied these techniques to problems with the relatively complete response property, where the defender's problem could always be solved, regardless of the decisions the attacker makes. Intuitively, these methods allow for the attacker to account

for how the defender has previously reacted to their decisions, resulting in better informed decisions by the attacker. These methods are improved upon in Yue et al. [129], applying these methods to problems that do not satisfy this property. The intuitive extension with these methods is that only the previous defender decisions that are relevant to the current attacker decision are accounted for. While this starts to exploit information from previous defender decisions, we can gain more information. When we can utilize partial information from the defender decisions that are infeasible with respect to the current interdiction decision, we can gain unique information that may not be accounted for in the previous feasible defender decisions. We extend previous C&CG methods to utilize partial information from previously identified defender decisions and, in particular, when a defender decision is not fully feasible with respect to the new attacker decision. Utilization of partial information from previous defender decisions can also be implemented in general solvers. The method of Lozano and Smith [72] similarly seeks to block previously identified defender decisions, where utilizing partial information would instead only block the infeasible components of previously identified defender decisions.

1.4 Organization of Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 introduces the problem of adding edges to enforce that a given vertex partition of a network is an optimal solution to the modularity maximization problem, as well as the problem of removing edges such that the resulting sub-network has the same vertex partition that maximizes modularity as the original network. We derive an integer programming framework to solve these problems, as well as heuristic approaches to find high quality solutions. These methods are tested on multiple networks, some popular in community detection and some popular in illicit network analysis. Chapter 3 introduces the max flow network interdiction problem with restructuring, where the defender is allowed to introduce new arcs to the network after interdiction decisions have been implemented. We model this problem as a bilevel mixed integer linear program, and improve upon column-and-constraint generation to solve this problem. This method is tested on network data simulating city-level drug smuggling networks, where we demonstrate our method vastly outperforms previous column-and-constraint generation methods, and allows us to provide insights into how restructuring impacts how we should allocate resources to disrupt these types of networks. Chapter 4 proposed a network gen-

erator for domestic sex trafficking networks to help address the data problem in applying OR tools to analyzing and disrupting human trafficking networks. Multiple sources have been triangulated to ensure that networks produced by the generator are realistic, including law enforcement case file analysis, interviews with domain experts, and a survivor-centered advisory group with first-hand knowledge of sex trafficking. We demonstrate the use of output networks in exploring policy recommendations from max flow network interdiction with restructuring, ie., the problem studied in Chapter 3. To do so, we propose a novel conceptualization of flow as the ability of a trafficker to control their victims. Chapter 5 introduces the multi-period max flow network interdiction problem with restructuring, an extension of the problem introduced in Chapter 3 to include a temporal component. In this problem, interdictions and restructurings are decided upon upfront, but implemented throughout the time horizon. We also introduce a variant of MMFNIP-R to include a second cooperating attacker that also has the ability to interdict victims and to prevent prospective victims from being recruited, modeling how social services would disrupt the network. We propose two augmentations to the column-and-constraint generation procedure to improve the solve time of the method. Chapter 6 concludes the dissertation and proposes avenues for future research regarding these topics.

CHAPTER 2
OPTIMIZING EDGE SETS IN NETWORKS TO PRODUCE GROUND TRUTH COMMUNITIES BASED ON MODULARITY

2.1 Introduction

In this chapter, we introduce two new problems regarding modularity maximization. The first problem seeks to minimize the number of edges added to the network such that a given community structure is the partition that maximizes modularity. The second problem seeks to find the sparsest sub-network such that this sub-network has the same partition that maximizes modularity as the original network. We propose a mixed integer nonlinear programming formulation for these problems, and devise a row generation algorithm and heuristic algorithms to solve these problems. We also provide examples of when adding an edge to the network change modularity contrary to how we would expect it to, and prove properties on how the inclusion of individual edge impacts modularity.

This chapter is organized as follows: in Section 2.2, our two problems of interest are formally introduced. In Section 2.3, we describe a row generation algorithm to solve the problem exactly, and reformulate the problem as a MILP. In Section 2.4, we will demonstrate properties of modularity that make it challenging to develop heuristics to solve this problem, and present our heuristic approaches. In Section 2.5, we present our computational results of our row generation algorithm and heuristics. We discuss our conclusions in Section 2.6.

2.2 Description of Problems

2.2.1 Edge Addition for Community Optimization

We first introduce the edge weight addition for community optimization problem. Consider a weighted network $G = (V, E)$ with weights $w : E \to \mathbb{R}$. Note that pairs of vertices that do not have an edge between them can be considered as having edges with weight 0. Let $\mathcal{P}$ be the set of feasible partitions of V, with ground truth partition $T = (C_1^T, \ldots, C_k^T) \in \mathcal{P}$.

This chapter previously appeared as: D. Kosmas, J.E. Mitchell, T.C. Sharkey, and B.K. Szymanski, *Optimizing edge sets in networks to produce ground truth communities based on modularity*, Networks, (2021) pp. 1-26. https://doi.org/10.1002/net.22084.

Let $\bar{E}$ be the set of edges that can have weight added to the edge between them, and let B_{ij} be the upper bound on the amount of weight that can be on edge (i,j). Let b_{ij} be the decision variable indicating how much weight is added to edge (i,j). This problem considers adding weights to G to enforce that the ground truth community structure $T = (C_1^T, \ldots, C_k^T)$ is a partition that maximizes the modularity of the new network and $w_{ij} + b_{ij} \leq B_{ij}$.

Edge Weight Addition for Community Optimization

Input: A network $G = (V, E)$ with weights $w : E \to \mathbb{R}$, set of edges that can have weight added to them $\bar{E}$, weight limits on edges $B : E \cup \bar{E} \to \mathbb{R}_{\geq 0}$ and a specified partition $T = \{C_1^T, \ldots, C_k^T\}$

Output: Set of weights b added to edges that enforce that T is a partition that maximizes the modularity of the network G with weights $w + b$.

Let $Q_P(w)$ be the modularity of G with edge weights w. We can model the edge weight addition for community optimization problem as the following:

$$\min_{b \in \mathbb{Z}_{\geq 0}} \sum_{(i,j) \in \bar{E}} b_{ij} \tag{2.1a}$$

$$\text{s.t. } T \in \arg\max_{P \in \mathcal{P}} \{Q_P(w+b)\} \tag{2.1b}$$

$$w_{ij} + b_{ij} \leq B_{ij} \text{ for all } (i,j) \in \bar{E} \tag{2.1c}$$

By enumerating over all possible partitions, we can easily convert the constraint $T \in \arg\max_{P \in \mathcal{P}} \{Q_P(b)\}$ in model (2.1) to the following equivalent model:

$$\min_{b \in \mathbb{Z}_{\geq 0}} \sum_{(i,j) \in \bar{E}} b_{ij} \tag{2.2a}$$

$$\text{s.t. } Q_T(w+b) \geq Q_P(w+b) \text{ for all } P \in \mathcal{P} \tag{2.2b}$$

$$w_{ij} + b_{ij} \leq B_{ij} \text{ for all } (i,j) \in \bar{E} \tag{2.2c}$$

In this program, the constraint $Q_T(w+b) \geq Q_P(w+b)$ enforces that the modularity associated with partition T after adding weights b is at least as large as the modularity of every other partition P. Thus, T is in the set of optimal partitions for the modularity maximization problem. To enforce that T is the unique optimal solution, the constraint could be replaced with $Q_T(w+b) \geq Q_{\mathcal{P}}(w+b) + \epsilon$ for some small $\epsilon > 0$.

In the case where G is unweighted, i.e., $w_{ij} = 1$ for all $(i,j) \in E$ and $w_{ij} = 0$ for all $(i,j) \notin E$, we can derive an equivalent model for model (2.2) that is an integer linear program. In this case, $\bar{E}$ will be the set of edges that can be added to the network, and $Q_P(E')$ will be the modularity of the partition P in the network $G' = (V, E \cup E')$, where $E' = \{(i,j) \in \bar{E} : b_{ij} = 1\}$. This special case can be modeled as:

$$\min_{E' \subseteq |\bar{E}|} |E'| \tag{2.3a}$$

$$\text{s.t. } Q_T(E') \geq Q_P(E') \text{ for all } P \in \mathcal{P} \tag{2.3b}$$

Since we can derive an integer linear program equivalent to model (2.3), which can be solved by commercial solvers, we will use this model as the basis for derivations in Section 2.3.

2.2.2 Edge Removal for Community Preservation

We now introduce the edge removal for community preservation problem. Consider a network $G = (V, E)$, and let P^* be a partition that maximizes modularity of G. The edge removal for community preservation problem seeks to identify the smallest set $E' \subseteq E$ such that P^* is a partition that maximizes modularity of $G' = (V, E')$.

Edge Removal for Community Preservation

Input: A network $G = (V, E)$, where P^* is a partition that maximizes the modularity of G

Output: A network $G' = (V, E')$, where $E' \subset E$ and P^* is a partition that maximizes the modularity of G'

This problem can be modeled similarly to model (2.3). We can express this as the following:

$$\min_{E' \subseteq E} |E'| \tag{2.4a}$$

$$\text{s.t. } Q_{P^*}(E') \geq Q_P(E') \text{ for all } P \in \mathcal{P} \tag{2.4b}$$

Unlike model (2.3), where we optimize over an extended set of edges $\bar{E}$, here we optimize over the original set of edges E. Since we are trying to maintain that the original optimal

partition P^* is a partition that maximizes modularity, we use P^* as the ground truth partition T in the constraints. Because the constraints in both models are the same, we can use a similar solution method to solve both problems.

2.3 Integer Programming Framework

2.3.1 Initial Framework

To solve model (2.3) by enumeration, we would need to verify the modularity of every possible partition, the number of which is exponentially large in the number of nodes. However, we expect that most of those partitions can be rejected based on modularity properties. For example, if every node is in the same cluster, or every node is in their own cluster, the modularity will always be zero. As long as our given partition has a non-negative modularity, we do not need to compare against these two partitions.

We propose a row generation method to solve these problems. In this framework, we iteratively identify partitions of higher modularity than our given modularity. We then use these partitions to determine what edges should be added to the network so the given partition has a higher modularity than these partitions.

Let $\mathcal{P}^k$ be the set of partitions identified by the k^{th} iteration of our iterative procedure. Instead of solving model (2.3), we solve the following:

$$\min_{z \in \{0,1\}^{|\bar{E}|}} \sum_{(i,j) \in \bar{E}} z_{ij} \tag{2.5a}$$

$$\text{s.t. } Q_T(z) \geq Q_{\mathcal{P}}(z) \text{ for all } P \in \mathcal{P}^k \tag{2.5b}$$

In model (2.5), we enforce that T has a modularity at least as large as every partition in $\mathcal{P}^k$, providing a lower bound on the objective value of model (2.3). However, this does not enforce that the given partition T is in the set of partitions that maximize modularity. Let z^k be the optimal solution found in model (2.5), and let E^k be the set of edges added to the network with solution z^k, i.e., $E^k = \{(i,j) \in \bar{E} : z^k_{ij} = 1\}$. We now find a partition that maximizes modularity of the new network with edges added. Let $\bar{A}$ be the adjacency matrix of $G^k = (V, E \cup E^k)$, $\bar{m} = |E \cup E^k|$, and $\bar{d}_i$ be the degree of node i in G^k. We find a partition that maximizes modularity with the following:

$$\max_{x\in\{0,1\}^{|V|^2}} \frac{1}{2\bar{m}} \sum_{i,j\in V} \left(\bar{A}_{ij} - \frac{\bar{d}_i \bar{d}_j}{2\bar{m}}\right)(1 - x_{ij}) \tag{2.6a}$$

$$\text{s.t. } x_{ij} \leq x_{ik} + x_{kj} \text{ for all } i, j, k \in V \tag{2.6b}$$

By solving model (2.6), we find a partition $\bar{P}$ that maximizes modularity of the new network, as well as its modularity. If $Q_T(z^k) = Q_{\bar{P}}(z^k)$, then we have that T is in the set of optimal partitions, and we are done. If not, then we have found a partition, $\bar{P}$, which has greater modularity than T which must be considered when adding edges. We set $\mathcal{P}^{k+1} = \mathcal{P}^k \cup \bar{P}$, and repeat this process. Formalizing this procedure yields Algorithm 1.

Algorithm 1 Row Generation for Edge Addition

Initialize: network $G = (V, E)$, ground truth partition T, initial set of partitions $\mathcal{P}^1$, set of includable edges $\bar{E}$, iteration counter $k = 1$.
while not converged **do**
  Step 1. Solve model (2.5) for z^k, E^k.
  if (2.5) is infeasible **then**
    terminate; model (2.3) is infeasible.
  end if
  Step 2. Create test network $G^k = (V, E \cup E^k)$. With test network G^k, solve model (2.6) for optimal partition $\bar{P}$.
  if $Q_T(z^k) = Q_{\bar{P}}(z^k)$ **then**
    terminate; E^k is the optimal solution to model (2.3).
  else
    Set $\mathcal{P}^{k+1} = \mathcal{P}^k \cup \bar{P}$, $k \leftarrow k + 1$, and return to Step 1.
  end if
end while

To analyze, each iteration of this procedure can result in one of three outcomes. First, if model (2.5) is infeasible, then model (2.3) is also infeasible. If model (2.5) is feasible, we have two cases. If $Q_T(z^k) = Q_{\bar{P}}(z^k)$ in the k^{th} iteration, we have found an optimal set of edges to add to the network. Otherwise, we have $Q_T(z^k) < Q_{\bar{P}}(z^k)$, so we include $\bar{P}$ in $\mathcal{P}^{k+1}$, and we continue the process. Since $|\mathcal{P}|$ is finite, our framework has a finite number of iterations, and will thus solve model (2.3) in finite time.

We note that modularity maximization is known to be NP-Hard, meaning solving model (2.6) in Algorithm 1 involves solving an NP-Hard problem each iteration. It may, however, not be necessary to solve model (2.6) to optimality; the purpose of solving model (2.6) is to determine if the partition T is optimal in G^k. If it is not optimal, we wish to find

a partition $\bar{P}$ with $Q_{\bar{P}}(z^k) > Q_T(z^k)$, not necessarily a partition that maximizes modularity of G^k. As long as a partition with $\bar{P}$ with $Q_{\bar{P}}(z^k) > Q_T(z^k)$ is identified, Algorithm 1 will still converge to the true solution. Alternatively, solving model (2.6) may be replaced with other methods, such as a heuristic method for identifying partitions with near-optimal modularity. In doing so, Algorithm 1 provides a lower bound on the objective value, since not solving model (2.6) to optimality may result in claiming that no partition exists such that $Q_P(z^k) \geq Q_T(z^k)$ even when such a partition might exist. However, for the networks we test our methods on, it was found that solving model (2.5) was more computationally challenging than solving model (2.6). We elaborate further on computational testing in Section 5.

While we choose to use modularity as our measure of community structure, note that our framework can work for any optimization-based clustering measure. In model (2.3), and likewise model (2.5), the constraint enforcing that the given partition T has modularity at least as large as that of all other partitions would be replaced with the desired measure of T being at least as large as that of all other partitions, and model (2.6) would be replaced with finding a partition that optimizes that measure. In this chapter, though, we are able to express model (2.3) as an integer program, rather than a nonlinear integer program, which we demonstrate over the next two subsections.

2.3.2 Nonlinear Integer Programming Representation

We now express model (2.3) as an integer program. We can rewrite the modularity of a given partition with added edges as:

$$Q_P(z) = \sum_{i,j \in V} \left(\frac{A_{ij} + z_{ij}}{2(m + \sum_{(s,t)\in \bar{E}} z_{st})} - \frac{(d_i + \sum_{l \in V} z_{il})(d_j + \sum_{h \in V} z_{jh})}{(2(m + \sum_{(s,t)\in \bar{E}} z_{st}))^2} \right) (1 - x^P_{ij}). \tag{2.7}$$

We can thus express the nonlinear integer program as:

$$\min_{z \in \{0,1\}^{|\bar{E}|}} \sum_{(i,j)\in \bar{E}} z_{ij} \tag{2.8a}$$

$$\begin{aligned} \text{s.t.} \sum_{i,j \in V} \left(\frac{A_{ij} + z_{ij}}{2(m + \sum_{(i,j)\in \bar{E}} z_{ij})} - \frac{(d_i + \sum_{l \in V} z_{il})(d_j + \sum_{h \in V} z_{jh})}{(2(m + \sum_{(s,t)\in \bar{E}} z_{st}))^2} \right) (1 - x^T_{ij}) \\ \geq \sum_{i,j \in V} \left(\frac{A_{ij} + z_{ij}}{2(m + \sum_{(i,j)\in \bar{E}} z_{ij})} - \frac{(d_i + \sum_{l \in V} z_{il})(d_j + \sum_{h \in V} z_{jh})}{(2(m + \sum_{(s,t)\in \bar{E}} z_{st}))^2} \right) (1 - x^P_{ij}) \end{aligned} \tag{2.8b}$$

for all $P \in \mathcal{P}$

We note that model (2.4) can be expressed similarly, where z_{ij} is no longer a binary vector indicating inclusion of edges, but a vector of integers, expressing the amount of weight added to edges.

2.3.3 Linearizing the Problem

The constraints in model (2.8) are nonlinear, but, by using standard techniques, we can create linear constraints that are equivalent. Below, we describe the process of linearizing these constraints, which we summarize in Theorem 1. The equivalent constraint in the weighted version of the problem can be linearized similarly, but we will highlight some key differences after the derivation is complete.

The first point to note is that the $2(m+\sum_{(i,j)\in\bar{E}} z_{ij})$ in the denominator is the same on both sides of the equality. Furthermore, it will always be strictly greater than 0, assuming the network has a nonempty edge set. Thus, multiplying both sides of the inequality by this denominator does not change the inequality, resulting in:

$$\begin{aligned}&\sum_{i,j\in V}\left(A_{ij}+z_{ij}-\frac{(d_i+\sum_{l\in V} z_{il})(d_j+\sum_{h\in V} z_{jh})}{2(m+\sum_{(s,t)\in\bar{E}} z_{st})}\right)(1-x^T_{ij})\\ \geq &\sum_{i,j\in V}\left(A_{ij}+z_{ij}-\frac{(d_i+\sum_{l\in V} z_{il})(d_j+\sum_{h\in V} z_{jh})}{2(m+\sum_{(s,t)\in\bar{E}} z_{st})}\right)(1-x^P_{ij}).\end{aligned} \tag{2.9}$$

Again, using that $2(m+\sum_{(i,j)\in\bar{E}} z_{ij}) > 0$, we multiply both sides of the inequality by $2(m+\sum_{(i,j)\in\bar{E}} z_{ij})$ to remove fractions from our inequality. Additionally, the factors $1-x^T_{ij}$ and $1-x^P_{ij}$ are multiplied by the same terms. Thus, we can equivalently express the inequality as the following:

$$\sum_{i,j\in V}\left((A_{ij}+z_{ij})(2(m+\sum_{(s,t)\in\bar{E}} z_{st}))-(d_i+\sum_{l\in V} z_{il})(d_j+\sum_{h\in V} z_{jh})\right)(x^P_{ij}-x^T_{ij})\geq 0 \tag{2.10}$$

We now have a quadratic constraint equivalent to the original constraint. To linearize this constraint, we use the McCormick inequalities [84]. For a pair of edges (i,j) and (s,t), let $w_{ijst} = z_{ij}z_{st}$. The following inequalities enforce that w_{ijuv} takes the correct value:

$$w_{ijst} \leq z_{ij}, \tag{2.11}$$

$$w_{ijst} \leq z_{st}, \tag{2.12}$$

$$w_{ijst} \geq 0, \tag{2.13}$$

$$w_{ijst} \geq z_{ij} + z_{st} - 1. \tag{2.14}$$

By multiplying out the quadratic terms, and replacing any bilinear terms with the appropriate variable, we can express the term multiplied by x_{ij}^T (and x_{ij}^P) as

$$\begin{aligned} f_{ij}(z,w) = 2mA_{ij} + 2A_{ij} \sum_{(s,t)\in\bar{E}} z_{st} + 2mz_{ij} + 2 \sum_{(s,t)\in\bar{E}} w_{ijst} \\ - \left(d_i d_j + d_i \sum_{h\in V} z_{jh} + d_j \sum_{l\in V} z_{il} + \sum_{h,l\in V} w_{iljh} \right). \end{aligned} \tag{2.15}$$

We can now express the linearized constraint as:

$$\sum_{i,j\in V} (x_{ij}^P - x_{ij}^T) f_{ij}(z,w) \geq 0 \tag{2.16}$$

Since the z variables are binary, we will have that $w_{ijst} = z_{ij} z_{st}$. Thus, our integer programming formulation is:

$$\min_{z\in\{0,1\}^{|\bar{E}|}, w} \sum_{(i,j)\in\bar{E}} z_{ij} \tag{2.17a}$$

$$\text{s.t.} \sum_{i,j\in V} (x_{ij}^P - x_{ij}^T) f_{ij}(z,w) \geq 0 \text{ for all } P \in \mathcal{P} \tag{2.17b}$$

$$w_{ijst} \leq z_{ij} \text{ for all } (i,j),(s,t) \tag{2.17c}$$

$$w_{ijst} \leq z_{st} \text{ for all } (i,j),(s,t) \tag{2.17d}$$

$$w_{ijst} \geq 0 \text{ for all } (i,j),(s,t) \tag{2.17e}$$

$$w_{ijst} \geq z_{ij} + z_{st} - 1 \text{ for all } (i,j),(s,t) \tag{2.17f}$$

Theorem 1. Model (2.17) is equivalent to model (2.8): there is a one-to-one correspondence between solutions for (2.8) and model (2.17).

Proof. We show that model (2.3) is equivalent to model (2.17) by showing that a feasible

solution to one model has a corresponding feasible solution to the other model with the same objective value. Note that, since the objective function value of both models is the same, as long as z remains unchanged between the two models, the objective value of the solution will also be unchanged.

Suppose (z, w) is a feasible solution to model (2.17). Since the constraints of model (2.8) are equivalent to constraints (2.16), and (z, w) is a feasible solution to model (2.17), z is a feasible solution to model (2.8) with equivalent objective value. Likewise, suppose z is a feasible solution to model (2.8). For pairs of edges $(i, j), (s, t)$, compute $w_{ijst} = z_{ij}z_{st}$. Suppose, without loss of generality, $z_{ij} = 0$, and thus $w_{ijst} = 0$. The McCormick inequalities will be satisfied, as $0 = w_{ijst} = z_{ij} \leq z_{st}$ and $w_{ijst} = 0 = z_{ij} \geq z_{ij} + z_{st} - 1$. Now suppose $z_{ij} = z_{st} = 1$. The McCormick inequalities are again satisfied, since $w_{ijst} = 1 \leq z_{ij} = z_{st}$ and $w_{ijst} = 1 \geq 1 = z_{ij} + z_{st} - 1$. Additionally, since the constraints of model (2.8) are equivalent to constraints (2.16), (z, w) is a feasible solution to model (2.17) with equivalent objective value. Thus, model (2.8) is equivalent to model (2.17). □

In the framework of Algorithm 1, we would solve model (2.17) to determine E^k, then solve model (2.6) to determine a partition that maximizes modularity of the network $G^k = (V, E \cup E^k)$. We additionally add a constraint regarding the previous iteration's objective value to model (2.17). Let c^k be the objective value in the k^{th} iteration. We add the constraint $\sum_{(i,j)\in K_{|V|}\setminus E} z_{ij} \geq c^k$, with c^0 initialized as 0. By enforcing that the objective value is non-decreasing, we can reduce the time of the branch-and-bound procedure needed to solve iterations where the objective value stays the same as in the previous iteration.

We note that, when extending these manipulations to the weighted version of the problem, we use the generalized McCormick envelope [84]. For integer variables z_{ij}, z_{st} with $l_{ij} \leq z_{ij} \leq u_{ij}$ and $l_{st} \leq z_{st} \leq u_{st}$, we can represent w_{ijst} as:

$$w_{ijst} \leq l_{st}z_{ij} + u_{ij}z_{st} - l_{st}u_{ij} \tag{2.18}$$

$$w_{ijst} \leq l_{ij}z_{st} + u_{st}z_{ij} - l_{ij}u_{st} \tag{2.19}$$

$$w_{ijst} \geq l_{st}z_{ij} + l_{ij}z_{st} - l_{ij}l_{st} \tag{2.20}$$

$$w_{ijst} \geq u_{st}z_{ij} + l_{ij}z_{st} - u_{ij}u_{st} \tag{2.21}$$

However, for non-binary z_{ij} and z_{st}, we no longer have the guarantee that $w_{ijst} = z_{ij}z_{st}$, and these substitutions will not result in an equivalent model. A potential solution to this

is to use binary expansion, replacing each integer variable with a set of binary variables to represent any value that integer variable can take [98]. More formally, for integer variable z_{ij} with $z_{ij} \leq u_{ij}$, we would create binary variables $b_{ij}^0, \ldots, b_{ij}^k$, with $k = \lfloor \log_2(u_{ij}) \rfloor$. We would then include $z_{ij} = \sum_{l=0}^{k} 2^l b_{ij}^l$, and replace z_{ij} with $\sum_{l=0}^{k} 2^l b_{ij}^l$ in the model, converting the bilinear terms with integer variables to bilinear terms with binary variables. While this substitution would result in an equivalent model, it also results in a large number of binary variables that does not scale well with problem size. Preliminary computational results show this is not a feasible approach.

2.3.4 Disjunctive Cuts and Additional Partitions

In order to better enforce that the ground truth partition is optimal, we augment model (2.5) by defining valid disjunctive cuts, and augment Algorithm 1 by generating additional partitions. We first consider how to enforce that a single node v is clustered correctly. For ease of notation, consider a pair of arbitrary partitions $P = (C_1 \cup \{v\}, C_2, C_3, \ldots, C_l)$ and $P' = (C_1, C_2 \cup \{v\}, C_3, \ldots, C_l)$, where P is the desired partition. Let $e_i(v)$ be the set of edges between v and cluster C_i for $i = 1, 2$ and let $e_-(v)$ be the set of edges between v and $\bigcup_{j=3}^{l} C_j$. Let $E(C_i, C_-)$ be the set of edges between C_i and $\bigcup_{j=3}^{l} C_j$, and let Q_- be the contribution to modularity by $C_3, \ldots, C_l$. We want to identify conditions on when $Q_P < Q_{P'}$ to derive valid inequalities that can be added to the model (2.17) to prevent it from occurring.

$$(4m^2)\,(Q_P - Q_{P'}) \tag{2.22}$$

$$\begin{aligned}
&= 4m(|E(C_1)| + |e_1(v)|) - (2|E(C_1)| + 2|e_1(v)| + |E(C_1, C_2)| + |e_2(v)| + |E(C_1, C_-)| + |e_-(v)|)^2 \\
&\quad +4m|E(C_2)| - (|2E(C_2)| + |E(C_1, C_2)| + |e_2(v)| + |E(C_2, C_-)|)^2 + 4mQ_- \\
&\quad -4m|E(C_1)| + (2|E(C_1)| + |E(C_1, C_2)| + |e_1(v)| + |E(C_1, C_-)|)^2 - 4mQ_- \\
&\quad -4m(|E(C_2)| + |e_2(v)|) + (2E(C_2) + 2|e_2(v)| + |E(C_1, C_2)| + |e_1(v)| + |E(C_2, C_-)| + |e_-(v)|)^2
\end{aligned} \tag{2.23}$$

$$\begin{aligned}
&= 4m(|e_1(v)| - |e_2(v)|) + (2|E(C_1)| + |e_1(v)| + |E(C_1, C_2)| + |E(C_1, C_-)|)^2 \\
&\quad -[(2|E(C_1)| + |e_1(v)| + |E(C_1, C_2)| + |E(C_1, C_-)|) + (|e_1(v)| + |e_2(v)| + |e_-(v)|)]^2 \\
&\quad - (|2E(C_2)| + |e_2(v)| + |E(C_1, C_2)| + |E(C_2, C_-)|)^2 \\
&\quad +[(2|E(C_2)| + |e_2(v)| + |E(C_1, C_2)| + |E(C_2, C_-)|) + (|e_1(v)| + |e_2(v)| + |e_-(v)|)]^2
\end{aligned} \tag{2.24}$$

$$\begin{aligned}
&= 4m(|e_1(v)| - |e_2(v)|) - (|e_1(v)| + |e_2(v)| + |e_-(v)|)^2 \\
&-2\left(2|E(C_1)| + |e_1(v)| + |E(C_1, C_2)| + |E(C_1, C_-)|\right)\left(|e_1(v)| + |e_2(v)| + |e_-(v)|\right) \\
&+2\left(2|E(C_2)| + |e_2(v)| + |E(C_1, C_2)| + |E(C_2, C_-)|\right)\left(|e_1(v)| + |e_2(v)| + |e_-(v)|\right) \\
&+(|e_1(v)| + |e_2(v)| + |e_-(v)|)^2
\end{aligned} \tag{2.25}$$

$$\begin{aligned}
&= (4m - 2|e_1(v)| - 2|e_2(v)| - 2|e_-(v)|)(|e_1(v)| - |e_2(v)|) \\
&-2(2|E(C_1)| + |E(C_1, C_-)| - 2|E(C_2)| - |E(C_2, C_-)|)(|e_1(v)| + |e_2(v)| + |e_-(v)|)
\end{aligned} \tag{2.26}$$

Note that $0 \leq |e_1(v)| + |e_2(v)| + |e_-(v)| \leq m$, so if $|e_1(v)| \leq |e_2(v)|$ and $2|E(C_1)| + |E(C_1, C_-)| > 2|E(C_2)| + |E(C_2, C_-)|$, then $Q_P < Q_{P'}$. Thus, we must have that

$$|e_1(v)| \geq |e_2(v)| + 1 \text{ or } 2|E(C_2)| + |(E(C_2, C_-)| \geq 2|E(C_1)| + |(E(C_1, C_-)|. \tag{2.27}$$

Likewise, if $|e_1(v)| < |e_2(v)|$ and $2|E(C_1)| + |E(C_1, C_-)| \geq 2|E(C_2)| + |E(C_2, C_-)|$, then $Q_P < Q_{P'}$. This results in needing to enforce

$$e_1(v)| \geq |e_2(v)| \text{ or } 2|E(C_2)| + |(E(C_2, C_-)| \geq 2|E(C_1)| + |(E(C_1, C_-)| + 1. \tag{2.28}$$

To define these cuts regarding T, we must first discuss how to determine which nodes are "classified incorrectly," especially when our ground truth community structure and the current optimal partition have a different number of clusters. Let $\bar{P}$ be a partition that maximizes modularity, as determined by Algorithm 1. We construct a complete bipartite network, where one set of nodes represents the clusters in T, and the other set represents the clusters in $\bar{P}$. For a pair of clusters C_i^T and $C_j^{\bar{P}}$, we set the edge weight for (i, j) to be $|C_i^T \cap C_j^{\bar{P}}|$, the number of nodes shared by those clusters. We then solve the maximum weight matching problem on this bipartite network, indicating how the clusters should be ordered. From this, we can easily determine which nodes are incorrectly classified. Note that if an optimal partition has more clusters than the ground truth community structure, then at least one cluster in an optimal partition will not be matched with any cluster in the ground truth. In this scenario, all nodes in this cluster are considered incorrectly classified. Similarly, if the ground truth community structure has more clusters than an optimal partition, at least one cluster in the ground truth communities will not be matched with any cluster in an optimal partition. All nodes in this unmatched cluster will also be considered incorrectly classified. An example of this distinction being important is when two clusters in the ground truth

community structure are merged into a single cluster in an optimal partition. While these nodes will all be clustered with all of the nodes they should be clustered with, whichever ground truth cluster is smaller will have its nodes considered incorrectly classified. This indicates that edges (or weights) must be added to distinguish this smaller cluster as its own individual cluster.

Once we have determined the set of incorrectly classified nodes M, we can now define the disjunctive cuts. Let $v \in M$ be an incorrectly classified node, and, without loss of generality, assume $v \in C_1^T$. Let $\bar{C}$ be the cluster v belongs to in $\bar{P}$. Suppose $\bar{C} \setminus C_1^T$ is nonempty, and consider $w \in \bar{C} \setminus C_1^T$. Without loss of generality, assume $w \in C_2^T$. We define the partition $P' = (C_1^T \setminus \{v\}, C_2^T \cup \{v\}, C_3^T, \ldots, C_l^T)$. By utilizing $\bar{P}$, we ensure that the partition generated provides a meaningful contribution by clustering v with a node with which it was previously clustered. Besides defining the disjunctive cuts, we also include the partition P' in $\mathcal{P}^k$. Computationally, this inclusion improved solve time.

In the case when $\bar{C} \setminus C_1^T = \emptyset$, the above partition is no longer properly defined, as $\bar{C} \subseteq C_1^T$. One option would be to have v in a cluster on its own in P'. This partition, however, is known to not be optimal as long as $d_v > 0$ [19]. Instead, we consider a partition that separates all of $\bar{C}$, not just a single node, since $\bar{C} \subset C_1^T$. We can define a new partition $P' = (C_1^T \setminus \bar{C}, \bar{C}, C_2^T, \ldots, C_l^T)$. This partition is well defined, since if $C_1^T \setminus \bar{C} = \emptyset$ and $\bar{C} \setminus C_1^T = \emptyset$, then $C_1^T = \bar{C}$ and thus $v \notin M$. By including the constraint $Q_T \geq Q_{P'}$, we enforce that edges are added to merge $C_1^T \setminus \bar{C}$ and $\bar{C}$.

At each iteration of Algorithm 1, we generate the set of incorrectly classified nodes M, and remove nodes whose cluster in $\bar{P}$ is a subset of a cluster in T. For each of these nodes v, we pick an arbitrary node w, where v and w are in the same cluster in $\bar{P}$, but different clusters in T, to define the disjunctive cuts and additional partition. Then, for every cluster $C_i^{\bar{P}} \in \bar{P}$, we check if it is a subset of a cluster in $C_j^T \in T$, and generate additional partitions for it when $C_i^{\bar{P}} \subset C_j^T$.

2.4 Challenges for Heuristic Approaches

The potential downsides of using model (2.17) include (i) that there could be a large number of binary variables, particularly if the network is sparse, and (ii) we must repeatedly solve model (2.17) as we generate new partitions to be added to it. With this, even small networks can be computationally expensive to solve, especially when trying to solve model

(2.4). Thus, good heuristics are important for obtaining high quality solutions in a reasonable amount of time. Before we present our heuristic algorithms, we discuss some counter-intuitive behaviors of modularity as a measure of community strength.

2.4.1 Counter-Intuitive Behavior of Modularity

As modularity is a measure of the strength of communities, there are two nice properties we would hope to have. Intuitively, we would expect that adding an edge between two nodes in the same cluster would increase the modularity score of that partition. Likewise, we would expect that adding an edge between two nodes in different clusters would decrease the modularity score of that partition. Neither of these intuitive properties are true. Figure 2.1 demonstrates that adding an edge within a cluster can decrease modularity.

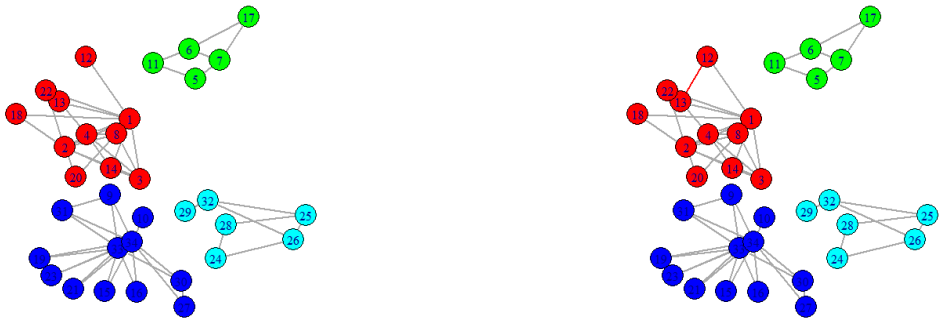


Figure 2.1: Example of adding an edge within a cluster decreasing modularity of the partition ©2021 John Wiley & Sons, Inc.

We construct this example by taking Zachary's karate club network [130] and removing all edges between distinct communities in an optimal partition of the network. The modularity of the partition where each disjoint component is its own community is 0.6753. By adding the red edge, which is between two nodes within the red cluster, the modularity decreases to 0.6724.

We can likewise create an example where adding an edge across clusters increases the modularity. Figure 2.2 demonstrates this behavior.

We create the example by starting with a complete network on 10 nodes. We then add six more nodes, and create two disjoint paths of edge-length two on these nodes. We take

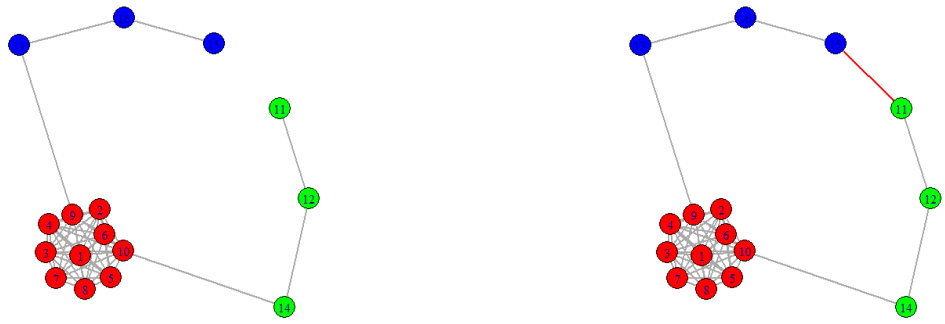


Figure 2.2: Example of adding an edge across clusters increasing modularity of the partition ©2021 John Wiley & Sons, Inc.

one endpoint of each path, and connect them to one node each in the complete sub-network. With the partition where the complete sub-network and two paths are in their own clusters, we have a modularity of 0.1424. By adding the red edge, which connects the degree 1 nodes in the blue and green clusters, the modularity increases to 0.1531.

A property of modularity being a global measure is that only the numbers of edges within clusters and between clusters are relevant to computing modularity, but not what specific nodes each edge connects [96]. We would then expect that the change in modularity of a particular partition from adding an edge is dependent only on the clusters the incident nodes are in, not the nodes themselves. Exact computations for the change in modularity are presented in Appendix A.

While this is a nice property for a global measure, this property makes developing heuristics based on local properties difficult; we cannot distinguish between edges to add to improve the modularity of our ground truth communities. Further, the change in modularity *across different partitions* will be different and thus understanding how adding an edge changes the current best partition remains challenging without enumerating all partitions. However, we still use our intuition in developing our heuristics.

We also wish to understand what edges may be necessary in creating a distinct cluster. From Brandes et al. [19], we know the following two properties.

Lemma 1. A partition P that maximizes modularity will not have disconnected clusters.

Lemma 2. A partition P that maximizes modularity will not have any clusters that consist of a single node with degree 1.

We can combine these two results into a more powerful result regarding star networks [99].

Theorem 2. Given a partition $P = \{C_1, \ldots, C_k\}$, a sparsest network, where each cluster is connected, to have P as a partition that maximizes modularity has each cluster C_i consisting of edges that form a star network.

Proof. Consider a network G on n nodes, and partition $P = \{C_1, \ldots, C_k\}$. In order to have cluster i constitute a connected component, we need to have at least $|C_i| - 1$ edges. Thus, to have that every cluster is a connected component, we need to have at least $n - k$ edges. We show that we can do so by demonstrating that if each cluster is a star network, then P is a partition that maximizes modularity.

Suppose the edges of G induce a star network on each cluster. Note that, by Lemma 1, nodes from distinct star networks will not be clustered together in an optimal partition. Furthermore, also by Lemma 1, in a partition that maximizes modularity, multiple degree 1 nodes in the i^{th} star network will not be clustered together unless they are also clustered with the degree $|C_i - 1|$ node. Thus, every center node in a star network will be in a distinct cluster, and every degree 1 node will be in its own cluster or with the center node of its star network. However, by 2, if there is a degree 1 node in its own cluster, that partition is not optimal. Thus, the only partition that maximizes the modularity of G will have the i^{th} cluster as the i^{th} star network. Since $|E(G)| = n - k$, there exists a network with $n - k$ edges that has P as a partition that maximizes modularity. □

This theorem helps us understand what edges would be useful to add to the network to get the best gain for the smallest number of edges. In particular, we will seek to 'reinforce' star-like sub-networks for each distinct cluster in the ground truth. This also provides a lower bound on the objective value of model (2.4).

2.4.2 Heuristic Algorithm for Edge Addition

Since we know exactly how adding edges to the network will impact modularity, we can easily describe the ideal edges to add: ones that are within a single cluster in ground truth partition T, and between two clusters in the currently optimal partition P^*. Our procedure

focuses on adding edges (or weights) to the network that increase the modularity of T. The procedure is:

1. Identify the "most incorrectly classified" node u with an incident edge that can be added to the network
2. Determine edge $(u, v) \in \bar{E}$ to add to the network
3. Determine if T belongs to the set of optimal partitions for $G' = (V(G), E(G) \cup \{(u, v)\}$

From the computations on change in modularity in Appendix A, we expect that choosing such an edge will decrease the gap between Q_{P^*} and Q_T, which works towards enforcing that T is a partition that maximizes modularity. We describe the procedure for unweighted networks first, then discuss how the procedure can be modified for weighted networks.

Before we can choose the node that is "most incorrectly classified", we need to first determine which nodes are incorrectly classified. As with defining the disjunctive cuts, we create an auxiliary bipartite network based on the clusters of P^* and T, and solve the maximum weight matching problem on this network. To determine which nodes are incorrectly classified, we solve the maximum weight matching problem as described in Section 2.3.4. Let M be the set of incorrectly classified nodes.

We now choose which incorrectly classified node $u \in M$ will be incident to the added edge. We do so by comparing the number of nodes that u should be clustered with in T but is currently not in a partition P^* that currently maximizes modularity, to the number of nodes it should be clustered with. Let $C^P(u)$ is the set of nodes in the cluster of P containing u. We order the nodes in M in decreasing order, determined by the fraction

$$q_u = \frac{|C^T(u) \setminus C^{P^*}(u)|}{|C^T(u)| - 1}. \tag{2.29}$$

Unless $C^T(u) = \{u\}$, the denominator of this term will be strictly greater than 0, and thus the term is well defined. Brandes et al. [19] proves that, as long as there are no isolated nodes in the network, a partition that maximizes modularity will never have a cluster that is just a single node. Additionally, they show that an isolated node can be placed in any cluster (including its own) without impacting modularity. Thus, if a ground truth partition ever has a cluster that is a single node u, that problem will either be infeasible if $d_u > 0$, or that node can be placed in its own cluster in a partition that maximizes modularity, so it will never be incorrectly classified.

By looking at the proportion of two terms of q_u, we can differentiate between different size clusters. If there are two nodes i and j such that $|C^T(i) \setminus C^{P^*}(i)| = |C^T(j) \setminus C^{P^*}(j)|$, and $|C^T(i)| < |C^T(j)|$, then $\frac{|C^T(i) \setminus C^{P^*}(i)|}{|C^T(i)-1|} > \frac{|C^T(j) \setminus C^{P^*}(j)|}{|C^T(j)-1|}$. By normalizing the clusters by their size, we allow for smaller clusters to more easily distinguish themselves and be selected earlier in the heuristic. Algorithm 2 formalizes this procedure.

Algorithm 2 Identify Most Incorrectly Classified Nodes ($MostIncorrect(G, T, P^*)$)

Initialize: network $G = (V, E)$, partitions T, P^*.
Step 1. Create auxiliary bipartite network between clusters of T and P^*, and solve the maximum weight matching problem to determine set of incorrectly classified nodes M.
Step 2. Order $M = \{u_1, \ldots, u_{|M|}\}$ in descending order based on $\frac{|C^T(u) \setminus C^{P^k}(u)|}{|C^T(u)-1|}$. Return M.

Once we have determined which nodes to prioritize adding an edge to, we must then determine which edge should be added to the network. Consider $u \in M$. Our first priority is to add an edge (u, v) such that $v \in C^T(u)$ and $v \notin C^{P^*}(u)$. We expect that adding such an edge to G will increase the modularity of T while decreasing the modularity of P^*. Let $N(u) = \{j \in V : j \in C^T(u) \setminus C^{P^*}(u) \wedge \ (u, v) \in \bar{E}\}$. Intuitively, $N(u)$ is the set of all nodes that u should be clustered with but currently is not, and an edge incident to u can be added to the network. Adding an edge where $v \in N(u)$ achieves our desired goal.

If $N(u)$ is empty, then no such edge can be added to the network, and we consider the next node in M. If, for every $u \in M$, $N(u) = \emptyset$, no edge can be added such that it is within a cluster of T and between clusters of P^*. Since we cannot decrease the modularity of P^* without also decreasing the modularity of T, we additionally consider edges that increase the modularity of both T and P^*. In some cases, such as when some cluster of T is a subset of a cluster of P^*, the edge we add to the network will increase the modularity of T more than it increases the modularity of P^*.

Instead of considering $N(u)$, we consider $\tilde{N}(u) = \{j \in V : j \in C^T(u) \wedge \ (u, v) \in \bar{E}\}$. This allows us to add edges that are both within $C^T(u)$ and C^{P^*}, relaxing our restriction to include edges we expect to also increase the modularity of P^*. If, for every $u \in M$, $\tilde{N}(u) = \emptyset$, then no incorrectly classified node can have an incident edge added to the network. As such, we expect that every edge we can add to the network will *decrease* the modularity of T; we use this as termination criterion under which the problem may be infeasible.

Suppose that, for some $u \in M$, there exists a node $v \in N(u)$ (or $v \in \tilde{N}(u)$). Thus, an

edge incident to u can be added to the network that we expect will increase the modularity of T. We now must choose which of these edges to add to the network. Recall that the change in modularity by adding an edge to the network is independent of the nodes incident to the edge, and only dependent on the cluster those nodes belong to. Thus, for a fixed partition, all of the considered edges will have the same impact on the modularity of the partition. This motivates us to consider how the edge we choose to add will impact which partition is optimal in the network resulting from adding that edge.

We want to add an edge (u, v) such that, when we find an optimal partition P' of the resulting network $G' = (V, E \cup \{(u, v)\})$, u is in a different cluster of P' than P^*, moving u to be clustered with a node it should be classified with. However, if v is not sufficiently connected to the cluster it belongs to, adding (u, v) could result in the opposite change of clustering, moving v into a different cluster in the resulting network. To avoid such a scenario, we measure v's contribution to the change in modularity by adding (u, v), which we define to be δ_v. Multiplying by $(2m)^2(2m+2)^2$, we compute this as follows:

$$\begin{aligned}\delta_v = {} & (2m)^2(2m+2)A_{vv} - (2m)^2(d_v+1)^2 - (2m)(2m+2)^2A_{vv} + (2m+2)^2d_v^2 \\ & +2\left((2m)^2(2m+2)A_{uv} + (2m)^2(2m+2) - (2m)^2(d_u+1)(d_v+1)\right) \\ & -2\left((2m)(2m+2)^2A_{uv} + (2m+2)^2d_ud_v\right) \\ & +2\left(\sum_{u,v\neq y\in C}\left((2m)^2(2m+2)A_{vy} - (2m)^2(d_v+1)d_y - (2m)(2m+2)^2A_{vy} + (2m+2)^2d_vd_y\right)\right)\end{aligned} \tag{2.30}$$

$$\begin{aligned} & = (-8m^2-8m)A_{vv} - 8m^2d_v - 4m^2 + 8md_v^2 + 4d_v^2 \\ & +2\left((-8m^2-8m)A_{uv} + 8m^3 + 8m^2 - (4m^2)(d_ud_v + d_u + d_v + 1) + (4m^2+8m+4)d_ud_v\right) \\ & +2\left(\sum_{u,v\neq y\in C}\left((-8m^2-8m)A_{vy} - 4m^2(d_vd_y + d_y) + (4m^2+8m+4)d_vd_y\right)\right) \\ & +(-8m^2-8m)A_{vv} + (8m+4)d_v^2 - 8m^2d_v - 4m^2 \\ & +2\left((-8m^2-8m)A_{uv} + (8m+4)d_ud_v + 8m^3 + 4m^2 - 4m^2d_u - 4m^2d_v\right) \\ & +2\left(\sum_{u,v\neq y\in C}\left((-8m^2-8m)A_{vy} + (8m+4)d_vd_y - 4m^2d_y\right)\right)\end{aligned} \tag{2.31}$$

$$\begin{aligned} &= 16m^3 + 4m^2 + (-8m^2 - 8m)A_{vv} - 16m^2 d_v \\ &+2 \sum_{y \neq v, y \in C} \left((-8m^2 - 8m)A_{vy} + (8m+4)d_v d_y - 4m^2 d_y \right) \end{aligned} \tag{2.32}$$

We drop the $16m^3 + 4m^2$ term, since that will be constant across all choices of v, reducing our rule to

$$\begin{aligned} \delta_v &= (-8m^2 - 8m)A_{vv} - 16m^2 d_v + \\ 2 &\sum_{y \neq v, y \in C} \left((-8m^2 - 8m)A_{vy} + (8m+4)d_v d_y - 4m^2 d_y \right). \end{aligned} \tag{2.33}$$

We choose to take the node $v^* = \arg\min_{v \in N(u)} \delta_v$, as a smaller local change in modularity indicates that node v is better connected within its cluster. We formalize this procedure in Algorithm 3.

We note that our choice is closely related to the degree of each candidate node. Intuitively, this aligns with Theorem 2. By adding edges to the highest degree node within a cluster, we work towards creating a star sub-network in the cluster. For incorrectly classified nodes connected to many nodes in other clusters, we need to add more edges than just establishing a base framework.

Once we have added an edge to the network to create G', we now need to determine if T belongs to the set of partitions that maximize the modularity of G'. We do so by finding a partition P' that maximizes the modularity of G', and computing the modularity of T. If $Q_T = Q_{P'}$, then T belongs to the set of optimal partitions, and we can terminate. Otherwise, we repeat the procedure. We formalize the full procedure in Algorithm 4.

Extending this heuristic to weighted networks is quite simple. While computations of m and d_i change, the above computation remains the same. We additionally allow for weight to be added to edges already existing in the network, removing the condition $A_{uv} = 0$ from the definition of $N(u)$ and $\tilde{N}(u)$. In a weighted network, we note that we could have instead added weight to an edge until the partition changes. Even if we were to do so, this does not guarantee that an incorrectly classified node will be correctly classified in future iterations nor that an edge will not be re-visited by the heuristic in future iterations, as demonstrated in the example in Figure 2.3, where Figure 2.3a is the initial optimal partition, and Figure 2.3b is the ground truth partition. In this example, each edge has weight 1, and the weight limit for each edge is 10.

In this network, weight is added to the colored edges in Figure 2.3c. In the first

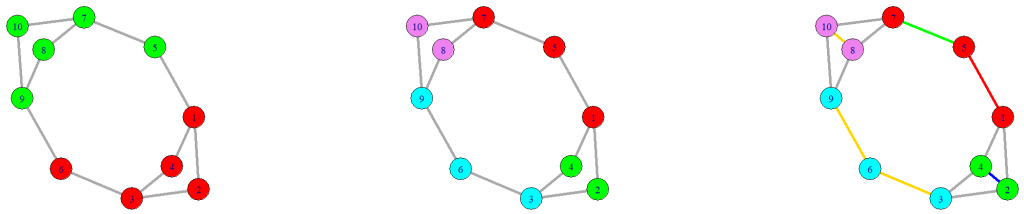
Algorithm 3 Choosing a Beneficial Edge ($ChooseEdge(G, T, M)$)

Initialize: network $G = (V, E)$, partition T, ordered set of incorrectly classified nodes $M = \{u_1, \ldots, u_{|M|}\}$, $chosen = false$, $l = 0$.
while $chosen = false$ **do**
  Step 1. Set $l \leftarrow l + 1$.
  if $l \leq |M|$ **then**
    Step 2a. Identify candidate nodes $N(u_l) = \{j \in V : C^T(u_l) \setminus C^{P^k}(u_l) \wedge (u_l, v) \in \bar{E}\}$.
    if $N(u_l) = \emptyset$ **then**
      Return to Step 1.
    else
      Step 3a. Set $u = u_l$.
      Step 3b. Compute δ_j for all $j \in N(u)$.
      Step 3c. Choose $j \in \arg\min_{i \in N(u)} \delta_i$. Return (u, v).
    end if
  else if $l > |M|$ and $l \leq 2|M|$ **then**
    Step 2b Set $\tilde{l} = l - |M|$
    Step 2c. Identify candidate nodes $\tilde{N}(u_{\tilde{l}}) = \{j \in V : C^T(u_{\tilde{l}}) \wedge (u_{\tilde{l}}, v) \in \bar{E}\}$.
    if $\tilde{N}(u_{\tilde{l}}) = \emptyset$ **then**
      Return to Step 1.
    else
      Step 4a. Set $u = u_{\tilde{l}}$
      Step 4b. Compute δ_j for all $j \in \tilde{N}(u)$.
      Step 4c. Choose $v \in \arg\min_{i \in \tilde{N}(u)} \delta_i$. Return (u, v).
    end if
  else
    Terminate, problem is infeasible.
  end if
end while

iteration, a unit of weight is added to the red edge, $(1, 5)$, causing nodes 1 and 5 to be clustered together. The second iteration adds the blue edge, $(2, 4)$. The third and fourth iterations add a unit of weight to the green edge $(5, 7)$. The addition of this weight causes nodes 5 and 7 to be clustered together, but separates node 1 from node 5. Our algorithm next adds another weight to the red edge. At the beginning, only one unit weight is needed for node 1 to be classified correctly. However, after more iterations, node 1 is no longer classified correctly, and an edge needs to be revisited.

Algorithm 4 Heuristic Algorithm for Edge Addition for Community Optimization

Initialize: initial network $G^1 = (V, E)$, set of edges added $E' = \emptyset$, ground truth partition T, iteration counter $k = 1$.
while not converged **do**
  Step 1. Find partition P^* that maximizes modularity of G.
  if $Q_T(G^k) = Q_{P^*}(G^k)$ **then**
    Terminate; T belongs to the set of optimal partitions. Return E'
  else
    Step 2. Set $M = MostIncorrect(G^k, T, P^*)$.
    Step 3. Set $(u, v) = ChooseEdge(G^k, T, M)$.
    Step 4. Set $G^{k+1} = (V(G^k), E(G^k) \cup (u, v))$, $E' \leftarrow E' \cup (u, v)$, $k \leftarrow k + 1$.
  end if
end while



(a) **(b)** **(c)**

Figure 2.3: Example network where an edge needs additional weight added after first visit by the heuristic approach where (a) is the original optimal partition, (b) is the desired optimal partition, and (c) highlights the edges that had added weight ©2021 John Wiley & Sons, Inc.

2.4.3 Heuristic Algorithm Edge Removal

When considering model (2.4), instead of selecting which edge to include from the edge set, we turn our attention to removing edges from the network. The idea behind this procedure is quite intuitive as well; order the edges in the network, then, one by one, check if removing the i^{th} edge results in a change in an optimal partition. If the optimal partition does not change, then that edge can be removed from the network. Note that, if the i^{th} edge was removed, that edge could have been critical for changing the optimal partition when removing any of the first $i-1$ edges. Thus, if any single edge is removed, we must check the full edge list again to determine if more edges can be removed. We repeat this procedure until it determines that removing any single edge in the network results in a change in an optimal partition. We formalize this approach in Algorithm 5.

Algorithm 5 Heuristic Algorithm for Edge Removal for Community Preservation

Initialize: initial network $G^1 = (V, E)$, set of edges removed $E' = \emptyset$, optimal partition T, $check = true$, iteration counter $k = 1$.
while $check = true$ **do**
  Step 1. Set $check = false$, $I = \emptyset$.
  Step 2. Order edges $E(G^k) = \{e_1, \ldots, e_{|E(G^k)|}\}$.
  for $i \in \{1, \ldots, |E(G^k)|\}$ **do**
    Step 3. Create network $G' = (V(G^k), E(G^K) \setminus (I \cup \{e_i\}))$.
    Step 4. Find an optimal partition P^* of G'.
    if $P^* = T$ **then**
      Step 4a. Set $I \leftarrow I \cup \{e_i\}$, $check = true$.
    end if
  end for
  Step 5. Create network $G^{k+1} = (V(G^k), E(G^k) \setminus I)$.
  Step 6. Set $E' \leftarrow E' \cup I$, $k \leftarrow k + 1$.
end while

We test the following rules for ordering the edges:

1. a prespecified order,

2. a random permutation,

3. a fixed order based on contribution to modularity,

4. a dynamically updated order based on contribution to modularity.

A weakness of the first rule is that the prespecified order may choose edges whose removal may cause fewer edges to be removed in subsequent iterations. The second rule randomizes the order in which the edges are processed in each iteration. This provides a workaround for the weakness of the first rule, but the quality of the solution obtained using this rule is determined by the orders randomly generated.

The third and fourth rules use an edge's contribution to modularity to determine order. We order the edges in decreasing order of their values in the modularity matrix, $M_{ij} = A_{ij} - \frac{d_i d_j}{2m}$. Intuitively, edges with a large value of M_{ij} have small products of degrees, meaning this edge has a low probability to be generated in the corresponding random network. We suspect that these edges are unlikely to be impactful in determining an optimal partition, making them good candidates for removal. For our third rule, we choose to pre-compute this ordering and use it in each iteration, as with the first rule. For the fourth rule, we update

this ordering after each iteration. Since it is possible that the removal of certain edges can cause changes in the importance of other edges to an optimal partition, the updates may result in more edges removed.

Since Algorithm 5 ensures that each subsequent network has the same optimal partition as the original network, we can improve the solution found by Algorithm 4 by taking the output solution G' as the input for Algorithm 5. This removes edges that were added, but later become redundant due to the addition of other edges. For this addition as post-processing, we simply choose to process the edges in the order in which they were added. Extending this algorithm to weighted networks is also straightforward. Instead of removing one edge at a time, we remove a unit of weight on one edge at a time.

2.5 Computational Results

We test our methods on multiple networks, some of which are popular in community detection literature and some popular in illicit network analysis. We retrieved this data from the CASOS Public Datasets [25] and Pajek Datasets [8]. Table 2.1 reports the networks we use for our experiments organized by the number of edges in the network. The top four networks are well known in the field of community detection, while the bottom six are known illicit networks. For the Italian Gangs, London Gangs, and Montreal Gangs data sets, which have disconnected components, we delete the nodes that are not in the largest connected component. While this is not a necessary modification, we do so to simplify the networks. We model the integer programming problem in AMPL with CPLEX 20.1 [57] as the solver. Experiments are conducted on a laptop with an Intel® Core™ i5-8250 CPU @ 1.6 GHz - 1.8 GHz and 16 GB RAM running Windows 10. We set a time limit of 7200 seconds. We develop the heuristics in R using the iGraph package [36]. Due to integer programming limitations, we additionally restrict the set of edges considered for addition in these experiments. To ensure feasibility, we start with the set of edges identified by the heuristics with post-processing and without edge restrictions as the initial set of allowable edges. We then generate the list of all pairs of nodes with shortest path length 2 between them, and randomly select from that list to add to the set of allowable edges. We consider 100 edges for each network.

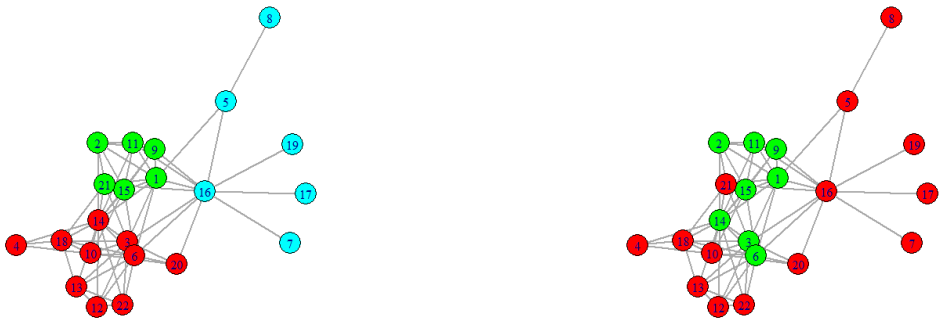
Table 2.1: Size of test networks

Network	$\|V\|$	$\|E\|$
Sawmill [86]	36	62
Dolphins small [73]	40	70
Karate [130]	34	78
Les Mis [64]	77	254
Ciel [89]	25	35
Caviar6 [89]	27	47
Rhodes [104]	22	66
Montreal Gangs [42]	29	75
Italian Gangs [25]	65	113
London Gangs [54]	48	133

2.5.1 Results for Edge Addition to Unweighted Networks

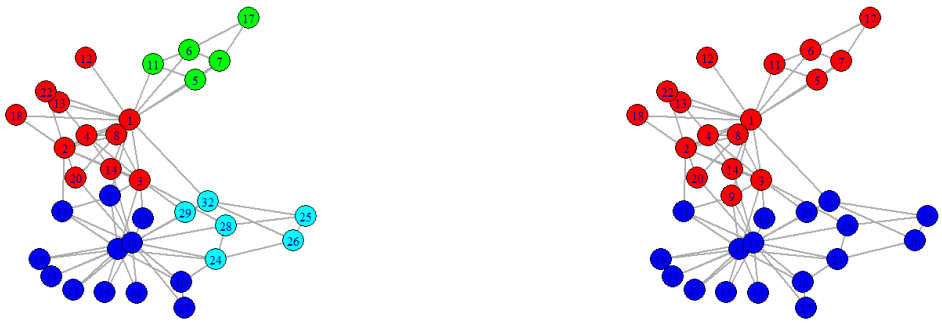
For networks with a known community structure, we use it as the ground truth. For the *Les Misérables* (Les Mis), London Gangs, and Italians Gangs data sets, we use the greedy clustering method [94] to identify a partition to be the ground truth. For the Ciel data set, we use the Louvain method [15] to identify a partition to be ground truth. Using the greedy clustering method resulted in partitions that had more differences in clusters than the Louvain method, except with the Ciel data set, where the greedy clustering method outputs an optimal partition. For networks that are originally weighted, we remove edge weights and experiment on their unweighted versions. Figures 2.4 and 2.5 demonstrate the differences in an optimal partition and the ground truth community structure for example networks.

Table 2.2 reports the initial number of incorrectly classified nodes as well as the number of edges added to each network to reach the desired communities. Table 2.3 reports the run time of each method. We denote the integer program without disjunctive cuts and extra partitions as IP, and the integer program with disjunctive cuts and additional partitions as IP+, and the number of partitions visited by our algorithm as nParts. Entries marked with * did not converge to the optimal solution within the time limit, and the lower bound on the optimal solution is reported. For instances where the integer programming formulation did not solve the problem to optimality, Table 2.4 reports the difference in modularity between the ground truth partition and an optimal partition after adding in the edges determined as the best known lower bound.



(a) (b)

Figure 2.4: Optimal partition (a) and ground truth community structure (b) of Rhodes network ©2021 John Wiley & Sons, Inc.



(a) (b)

Figure 2.5: Optimal partition (a) and ground truth community structure (b) of Karate Club network ©2021 John Wiley & Sons, Inc.

Table 2.2 shows that the integer programming formulation is able to find a relatively small number of edges compared to the density of the network in the case of the Karate club and Ciel networks. For some networks, such as Caviar6 and Montreal Street Gangs, the layout of the network requires edges to be added to distinguish a smaller cluster from a larger cluster that can relatively easily absorb it, as well as a significant number of edges that connect nodes that are clustered together in the ground truth, but do not have any edges in the between them in the network. However, for the other networks, the integer programming

Table 2.2: Comparison of number of edges added to tested networks with edge restrictions

Network	Init # Misclassified	#Edges Added (IP)	#Edges Added (IP+)	#Edges Added (Alg4)	#Edges Added (Alg4&5)
Sawmill	10	9	9	16	13
Dolphins small	17	24*	29*	34	30
Karate	12	8	8	11	9
Les Mis	16	18	18	21	21
Ciel	3	1	1	1	1
Caviar6	3	11	11	12	12
Rhodes	10	18*	18*	34	34
Montreal Gangs	16	18*	18*	45	42
Italian Gangs	21	2	2	4	4
London Gangs	8	5	5	7	7

Table 2.3: Run time (in seconds) on tested networks with edge restrictions

Network	IP (nParts)	IP+ (nParts)	Alg4	Alg4&5
Sawmill	309.08 (14)	3073.24 (8)	4.83	14.25
Dolphins small	7200* (8)	7200* (9)	26.47	258.66
Karate	537.84 (16)	72.63 (7)	3.80	13.82
Les Mis	507.69 (9)	255.67 (6)	690.38	1356.53
Ciel	1.67 (1)	1.67 (1)	0.09	0.13
Caviar6	126.81 (10)	36.53 (5)	4.95	10.68
Rhodes	7200* (9)	7200* (6)	4.69	7.31
Montreal Gangs	7200* (2)	7200* (2)	627.23	1256.49
Italian Gangs	33.34 (4)	41.98 (3)	42.02	66.89
London Gangs	66.17 (8)	43.11 (4)	26.23	42.14

formulation is not able to solve the problem within the time limit. This is due to the edge-anonymity property of modularity, as that unintentionally creates a lot of symmetry in the branch-and-bound procedure for the master problem. Integer programming problems with symmetry are particularly difficult to solve [97]. These symmetries prevented the branch-and-bound procedure from meaningfully increasing the lower bound on the objective value when fathoming nodes in the branch-and-bound tree. Additionally, as the number of edges being included increases, the number of symmetric solutions increases exponentially. We observed that, in early iterations and in subsequent iterations where the master problem objective value stays the same, the master problem is solved to optimality relatively quickly.

Table 2.4: Optimality gap in modularity for unsolved instances

Network	$Q_{P^*}(z^*)$ (IP)	$Q_T(z^*)$ (IP)	$Q_{P^*}(z^*) - Q_T(z^*)$ (IP)	$Q_{P^*}(z^*)$ (IP+)	$Q_T(z^*)$ (IP+)	$Q_{P^*}(z^*) - Q_T(z^*)$ (IP+)
Dolphins small	0.4845518	0.4586351	0.0259167	0.4759208	0.4502092	0.02571166
Rhodes	0.1748866	0.154691	0.02019558	0.1868622	0.154691	0.0321712
Montreal Gangs	0.2429761	0.128801	0.114175	0.2099087	0.1315759	0.07833276

However, in later iterations when the number of edges needed must be larger than in the previous iteration, the master problem may not necessarily solve to optimality due to the number of branches that need to be explored. We suspect that, in each iteration of the master problem, the integer program identifies the optimal solution with respect to the set of previously explored partitions, but does not complete the branching procedure to verify that the identified solution is indeed optimal. Appendix B presents the comparison in solve time between the master and subproblems, with the default symmetry breaking setting and maximum symmetry breaking setting.

We note that this issue is not solely based on the size of the network. The integer programming framework was able to solve the problem on the Les Mis, Italian Gangs, and London Gangs networks, but not on the Dolphins, Rhodes or Montreal Gangs networks. The former three networks all have more nodes and edges than the latter three networks. Additionally, the former three networks all had ground truth communities determined by using the greedy clustering procedure. This indicates that the difficulty of the problem is more dependent on the differences between the ground truth community and the initial optimal community structure than the size of the network.

Including disjunctive cuts and additional partitions improves the solve time of the integer programming formulation for the Karate club network, and only hinders the solve time of the Sawmill network. This indicates that the extra information generated may have prevented a useful partition that was visited by the base integer program from being visited by the augmented procedure. For the Dolphins network, the best known lower bound increases from 24 to 29, drastically improving the result, and improving the known quality of the heuristic solution. For Montreal Gangs and Rhodes, no improvements are made by including the disjunctive cuts and additional partitions. This indicates that the edge-anonymity caused the solver to stall early on in this procedure. Increasing the run time to 36000 seconds on these instances shows that little-to-no progress is made in improving these

lower bounds for either procedure. This demonstrates that problem difficulty is not only dependent on the size of the network, but the density of the network and difference between the ground truth community structure and the initial optimal partition.

The heuristic approach is able to find reasonably high quality solutions as compared to the integer program optimal solutions before post-processing is implemented. With post-processing, we are able to reduce the number of edges needed in some networks. Furthermore, where the integer program may not solve the problem in the time limit, the heuristic approach is able to find a solution in under an hour in all instances. When the integer program does solve the problem within the time limit, the heuristic approach is able to solve the problem significantly faster. We note that, since some of the allowable edges are determined by the heuristic without edge restrictions, adding the same edge restrictions to the heuristic will result in the same solution. However, allowing any edge currently not in the network to be added to the integer programming formulation would drastically increase the run time.

We observe that when the ground truth communities are relatively similar to the initial optimal community structure, fewer edges are needed. Additionally, if the ground truth community structure has clear boundaries between the communities, fewer edges are needed. Figures 2.4 and 2.5 help distinguish this difference. In Figure 2.5b, there are very few edges between the two clusters, and there are very few nodes that have edges that cross between clusters. This results in fewer edges being needed to properly cluster these boundary nodes. However, in Figure 2.4b, the distinction between the clusters is significantly less clear, resulting in a need for each cluster to have more edges added to properly distinguish them. Another example of this is the difference between the Caviar6 and Ciel networks. While both have only three initially incorrectly classified nodes, the Caviar6 network has a cluster that is mostly absorbed into another cluster, where the boundaries between the clusters in the Ciel network are more distinct. This causes a need for more edges in the Caviar6 network to properly distinguish this cluster.

2.5.2 Results for Edge Weight Addition to Weighted Networks

We also test our methods on weighted versions of each network used in our unweighted test set. Networks that are already weighted have an upper bound on the weight of edge set to the weight of the maximum weight edge. For networks that are unweighted, we randomly generate weights between 1 and 15 for each edge, with a maximum edge weight of

15 allowed. We use the unweighted optimal partition as the ground truth, and verify that an optimal partition for the weighted network is not the same as that of the unweighted network. We note that, unlike with unweighted networks, we cannot compare the heuristic results directly against the integer programming results. The integer program cannot find the optimal solution due to the inaccuracy in the McCormick inequalities. Table 2.5 reports the weights on networks used for experiments, using the same ordering as above.

Table 2.5: Total weights of edge sets of test networks

Network	Total Weight
Sawmill	481
Dolphins small	544
Karate	654
Les Mis	2030
Ciel	198
Caviar6	668
Rhodes	524
Montreal Gangs	621
Italian Gangs	915
London Gangs	980

As before, we restrict the set of edges to which weight can be added to in each network. We restrict this to be the set of currently existing edges, enforcing that no new edges can be added. Additionally, we restrict the total weight that an edge can have. Table 2.6 reports the initial number of incorrectly classified nodes and the amount of weight added to each network. Table 2.7 reports the run time of each method.

For all networks except Caviar6 and London Gangs, we are able to add less than 12% of the initial total weight to the network in order to return to the desired clustering. With Caviar6, we see that this percentage is much higher, at 17% of the initial total weight. This is likely due to the large disparity in distribution of edge weights; there are a few nodes with a lot of weight on incident edges, but many nodes with a small amount of weight on incident edges. Being able to correctly classify these nodes with a small amount of weight on incident edges takes much more weight. Similar to the unweighted case, the post-processing is able to reduce the total weight added by a small amount, but this decrease is not as significant as in the unweighted case. In the case of London Gangs, the heuristic determines the problem is infeasible with edge restrictions. The weights on this network were randomly generated, so

Table 2.6: Comparison of amount of weight added to tested networks with edge restrictions

Network	Init. #Misclassified	Weight Added (Alg4)	Weight Added(Alg2&3)
Sawmill	16	30	26
Dolphins small	8	61	58
Karate	13	45	44
Les Mis	11	48	47*
Ciel	3	12	12
Caviar6	7	115	113
Rhodes	6	38	31
Montreal Gangs	11	45	43
Italian Gangs	16	30	30
London Gangs	17	Infeasible*	Infeasible*

Table 2.7: Run time (in seconds) on tested networks with edge restrictions

Network	Alg2	Alg2&3
Sawmill	11.02	48.58
Dolphins small	20.79	68.19
Karate	46.15	120.11
Les Mis	2243.14	7200*
Ciel	0.51	0.96
Caviar6	17.42	60.38
Rhodes	2.64	7.37
Montreal Gangs	51.06	169.83
Italian Gangs	158.94	367.81
London Gangs	6073.99*	6073.99*

we know the heuristic is incorrect in this determination; by adding weights such that every edge in the network has the same weight, the optimal clustering will be the same as in the unweighted network. This highlights that adding weights in a greedy fashion may prevent the problem from being solved correctly.

We next report results on weighted networks without edge restrictions. Table 2.8 reports the initial number of incorrectly classified nodes, as well as the number of edges added to each network to reach the desired communities. Table 2.9 reports the run time of each method.

In this case, removing edge restrictions results in similar overall weight being added

Table 2.8: Comparison of amount of weight added to tested networks without edge restrictions

Network	Init. #Misclassified	Weight Added (Alg2)	Weight Added(Alg2&3)
Sawmill	16	26	22
Dolphins small	8	62	58
Karate	13	45	44
Les Mis	11	48	47*
Ciel	3	12	12
Caviar6	7	115	113
Rhodes	6	28	26
Montreal Gangs	11	48	46
Italian Gangs	16	31	30
London Gangs	17	151	134

Table 2.9: Run time (in seconds) on tested networks without edge restrictions

Network	Alg2	Alg2&3
Sawmill	9.67	36.08
Dolphins small	21.22	52.27
Karate	50.97	145.72
Les Mis	2482.99	7200*
Ciel	0.50	0.96
Caviar6	17.57	60.15
Rhodes	1.72	6.93
Montreal Gangs	34.32	90.14
Italian Gangs	187.92	762.48
London Gangs	4643.55	7200*

to the networks, and has run times comparable to that of the restricted edge case. An interesting result is that less weight needs to be added to the Dolphins and Montreal Gangs networks when edge restrictions are enforced. Recall that the proposed algorithms act in a greedy fashion by focusing on local decisions at each point. These results indicate that adding weight to any possible edge can result in adding weight to a locally good choice, but creates global difficulties in later iterations.

2.5.3 Results for Edge Removal

Here, we present the results of the edge removal for community preservation problem. The edge-anonymity prevented the integer programming formulation from making meaningful progress towards a solution, even with additional constraints enforcing that clusters are connected components. We present the number of edges remaining and run time on each network for all four rules in Table 2.10, with run time being reported in Table 2.11. For rule 2, we run 100 tests, and report the median, minimum and maximum of the number of edges remaining, and the mean and standard deviation of the run times. For the Les Mis data set, we only run 20 tests due to lengthy solve time. Additionally, we expect that our method will remove all cross-cluster edges; we expect removing them will increase the modularity of an optimal partition. For all networks tested, we verified that removing such edges resulted in the resulting network having the same optimal partition as the initial network, allowing us the apply our method to these reduced networks. We test our method on these reduced networks, with number of edges remaining presented in Table 2.12 and run times presented in Table 2.13. We report the number of edges remaining after removing cross-cluster edges as Pre, and the number of edges determined by Theorem 2 in the column LB.

Table 2.10: Comparison of number of edges remaining in tested networks

Network	#Edges	Rule 1	Rule 2 Med (Min, Max)	Rule 3	Rule 4	LB
Sawmill	62	46	34 (33, 46)	34	34	32
Dolphins small	70	53	43 (38, 68)	42	42	34
Karate	78	31	33 (30, 36)	31	31	30
Les Mis	254	137	74 (72, 140)	137	137	71
Ciel	35	26	26 (25, 28)	25	25	22
Caviar6	47	23	23 (23, 35)	23	23	23
Rhodes	66	21	23 (21, 27)	21	21	19
Montreal Gangs	75	35	32 (29, 36)	35	36	26
Italian Gangs	113	62	64 (62, 72)	64	64	60
London Gangs	133	44	45 (43, 88)	44	44	42

Here we see that for almost every data set, the rules ordering the edges based on contribution to modularity either perform as well as or better than the other rules. The minimums obtained by the second rule are at least as good as these rules, but the medians rarely outperform the other rules. Additionally, the high standard deviations in run time

Table 2.11: Run time (in seconds) on tested networks

Network	Rule 1	Rule 2 Mean (SD)	Rule 3	Rule 4
Sawmill	14.78	22.09 (8.40)	23.96	23.76
Dolphins small	98.06	35.17 (8.69)	24.67	23.89
Karate	27.11	26.41 (13.37)	23.06	23.09
Les Mis	7200*	5811.00 (1140.88)	7200*	7200*
Ciel	2.51	2.11 (0.46)	1.70	1.69
Caviar6	12.44	9.25 (2.90)	11.54	11.59
Rhodes	6.00	2.97 (0.84)	2.72	2.64
Montreal Gangs	47.93	111.50 (57.73)	172.28	177.58
Italian Gangs	837.89	1023.24 (235.80)	1128.15	1156.27
London Gangs	334.29	1310.91 (1377.77)	628.27	640.23

Table 2.12: Comparison of number of edges remaining in tested networks with pre-processing

Network	#Edges	Pre	Rule 1	Rule 2 Med (Min, Max)	Rule 3	Rule 4	LB
Sawmill	62	51	46	34 (33, 43)	34	34	32
Dolphins small	70	63	42	42 (38, 57)	42	42	34
Karate	78	57	31	33 (30, 40)	30	30	30
Les Mis	254	194	104	73 (72, 81)	95	95	71
Ciel	35	29	26	26 (25, 26)	25	25	22
Caviar6	47	27	23	23 (23, 23)	23	23	23
Rhodes	66	46	23	23 (21, 28)	21	21	19
Montreal Gangs	75	44	34	33 (29, 37)	33	33	26
Italian Gangs	113	87	68	68 (62, 73)	62	62	60
London Gangs	133	90	59	44 (43, 49)	44	44	42

make it less appealing, even when it is the fastest rule. The second rule only consistently outperforms the third and fourth rules in the Les Mis data set. This is likely due to being able to identify orderings that remove problematic edges that prevent the removal of more edges in later iterations. We note that the first rule only happens to work well when the ordering of the edges is already satisfactory. While it performs well here, it is not guaranteed to do so in general, such as with the Dolphins data set, where the third and fourth rule will be consistent regardless of initial edge ordering. We note that, on these networks, the dynamic reordering impacts execution time negligibly, but does not improve the results. Additionally, by applying pre-processing to remove edges that cross clusters, we can remove at least 10% of

Table 2.13: Run time (in seconds) on tested networks with pre-processing

Network	Rule 1	Rule 2 Mean (SD)	Rule 3	Rule 4
Sawmill	12.76	11.05 (1.39)	10.78	10.75
Dolphins small	18.04	20.63 (3.61)	19.95	19.97
Karate	8.78	8.87 (0.93)	10.03	9.86
Les Mis	1696.14	1253.12 (182.17)	2086.95	1777.45
Ciel	1.46	1.49 (0.13)	1.42	1.45
Caviar6	2.18	2.42 (0.37)	2.50	2.31
Rhodes	1.76	1.65 (0.30)	1.61	1.51
Montreal Gangs	11.75	12.25 (1.40)	11.29	11.74
Italian Gangs	308.31	349.39 (37.99)	285.99	295.70
London Gangs	59.17	67.19 (9.26)	68.42	71.46

the edges in each network while still maintaining the same optimal partition. This reduction significantly improves the solve time in every network. Additionally, applying our algorithm to these reduced networks consistently improved the quality of the solutions found, and decreased the range of solutions identified by the second rule.

2.6 Conclusion

We define two new problems regarding the modularity of a network. In the first problem, we seek to find the minimum set of edges (or weights) to add to a network to enforce that a given partition maximizes modularity. In the second problem, we seek to find the minimum set of edges in the network such that a partition optimizing modularity of the reduced network and the full network is the same. We provide an integer programming framework for these problems, and augment the method with the generation of disjunctive cuts and additional partitions. We demonstrate how the counter-intuitive behavior of modularity proves problematic for the branch-and-bound procedure and prevents proving nice properties about how adding edges impacts modularity. The integer programming formulation is able to solve the first problem when the ground truth communities are relatively distinct or similar to the initial optimal clusters, demonstrating that the difficulty of the problem is not just dependent on the size of the network. We also devise heuristics that are able to find high quality solutions to the problem significantly quicker than the integer programming formulation, and find solutions to the problem on instances that the integer programming formulation is

not able to solve. For the second problem, our heuristics find that we are able to identify a small number of edges necessary to maintain the original community structure, often within a small percentage of edges away from identifying star networks within each cluster.

CHAPTER 3
INTERDICTING RESTRUCTURING NETWORKS WITH APPLICATIONS IN ILLICIT TRAFFICKING

3.1 Introduction

In this chapter, we introduce a new class of max flow network interdiction problems where the defender is allowed to restructure arcs in the network after interdiction decisions have been implemented. We first describe some properties on how restructuring impacts the optimal flow. These observations yield important insights about how disrupting lower levels of drug trafficking or human trafficking networks may not impact their operations when replacing the disrupted participants is easy for the network. We then formulate a bilevel integer program (BIP) to model the problem. We provide a column-and-constraint generation (C&CG) algorithm to solve this class of problems, and compare the interdiction decisions to those of the standard max flow network interdiction problem. By utilizing partial information from previously explored restructuring decisions, we are able to solve this problem faster than previous C&CG algorithms.

This chapter is organized as follows: Section 3.2 formally introduces the max flow network interdiction problem with restructuring, describes properties on when restructuring will not increase the flow, and interprets these properties in the context of disrupting drug trafficking networks. Section 3.3 discusses how drug trafficking networks can be modeled as max flow network interdiction problems, and how we model when the network is able to restructure. Section 3.4 derives a column-and-constraint generation algorithm that utilizes partial information from previous iterations. Section 3.5 compares this algorithm to other column-and-constraint generation algorithms, and Section 3.6 analyzes the results on drug trafficking networks. Section 3.7 concludes the chapter.

This chapter has previously appeared as: D. Kosmas, T.C. Sharkey, J.E. Mitchell, K.L. Maass, and L. Martin, *Interdicting Restructuring Networks with Applications in Illicit Trafficking*, preprint, arXiv:2011.07093 [math.OC], 2020. Submitted to Eur. J. Oper. Res

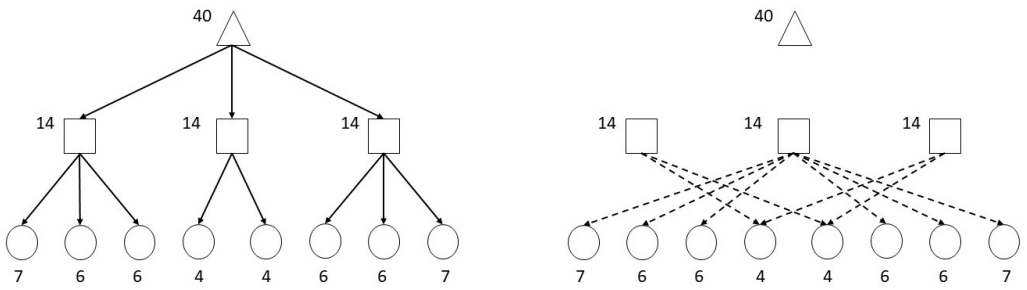
3.2 Problem Statement and Theoretical Policy Analysis

3.2.1 Problem Description and a Bilevel Integer Programming Formulation

The max flow network interdiction problem with restructuring (MFNIP-R) is played over three sets of decisions, one for the attacker then two for the defender. This game is played on a network $G = (N, A)$, where N is the set of nodes, A is the set of arcs initially in the network, and A^R is the set of restructurable arcs, which are arcs that can be added to the network. Nodes and arcs are assigned capacities $u : N \cup A \cup A^R \to \mathbb{R}_+$. In this chapter, we consider $A \cap A^R = \emptyset$. Flow is sent along the arcs from a source node α to a sink node ω. Without loss of generality, we assume there are no arcs into α and out of ω. First, the attacker chooses a set of nodes to disrupt in the network. Disrupting a node is a binary decision that sets the capacity of that node to 0. Second, the defender chooses a set of arcs from A^R to restructure, adding those arcs in the network. Restructuring an arc is a binary decision that changes that arc from having 0 capacity to its specified non-zero capacity. Lastly, the defender maximizes the flow through the restructured network. While we could consider the last two sets of decisions as a single set of decisions since they belong to the same decision-maker working towards the same goal in sequence, by considering them separately, we can develop a decomposition algorithm that leverages strong duality of linear programs. For a background in modeling max flow problems as linear programs, including converting problems with multiple source/sink nodes to single source/sink nodes, see [13].

Figure 3.1 displays a sample hierarchical network, which is how city-level drug trafficking networks have previously been modeled. Figure 3.1a is the original network, and Figure 3.1b is the set of restructurable arcs that may be added to the network. In the context of drug trafficking, the triangle node would be the supplier, the square nodes would be the drug dealers, and the circle nodes would be the drug users. In a max flow model, the supplier would be the source node, and the drug users would be sink nodes.

We note that, traditionally, max flow network interdiction models disrupt arcs instead of nodes. However, in previous work on disrupting city-level drug trafficking networks [9, 75, 109], their models disrupt nodes, since the interdiction decisions are modeling law enforcement disrupting the operations of individuals involved in the network, not communications or the passing of drugs between the individuals. Malaviya et al. [75] demonstrates the equivalency between interdicting nodes and interdicting arcs. Our results can be extended to more general problems that allow for both node and arc interdiction; however, we focus



(a) **(b)**

Figure 3.1: Example network motivated by city-level drug trafficking networks, where (a) is the current network and (b) is the set of restructurable arcs

on node interdictions given our motivating application.

We can model MFNIP-R as a bilevel integer program with integer decisions in both the upper and lower level problems. The upper level (attacker's) problem is the problem of finding the interdiction plan that minimizes the restructured flow, and the lower level (defender's) problem is the problem of finding the restructuring plan and maximum flow in the restructured network. Let x_{ij} be the variable representing the amount of flow on arc $(i,j) \in A$ and x_i be the amount of flow through node $i \in N$, y_i be the indicator of whether or not node $i \in N$ has been interdicted, and z_{ij} be the indicator of whether or not arc $(i,j) \in A^R$ has been restructured. More formally,

$$y_i = \begin{cases} 1 \text{ if node } i \text{ has been interdicted,} \\ 0 \text{ otherwise,} \end{cases} \tag{3.1}$$

and

$$z_{ij} = \begin{cases} 1 \text{ if arc } (i,j) \text{ has been restructured,} \\ 0 \text{ otherwise.} \end{cases} \tag{3.2}$$

Let Y be the set of all feasible interdiction decisions, and $Z(y)$ be the set of all feasible restructuring decisions with respect to interdiction decision $y \in Y$, and assume these sets are finite. The following BIP models the attacker's problem:

$$\min_{y \in Y} \max_{x,z} \sum_{(\alpha,i) \in A} x_{\alpha i} \tag{3.3a}$$

$$\text{s.t.} \sum_{(j,i) \in A \cup A^R} x_{ji} - x_i = 0 \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.3b}$$

$$x_i - \sum_{(i,j)\in A\cup A^R} x_{ij} = 0 \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.3c}$$

$$0 \leq x_{ij} \leq u_{ij} \text{ for } (i,j) \in A \tag{3.3d}$$

$$0 \leq x_i \leq u_i(1 - y_i) \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.3e}$$

$$0 \leq x_{ij} \leq u_{ij} z_{ij} \text{ for } (i,j) \in A^R \tag{3.3f}$$

$$z \in Z(y) \tag{3.3g}$$

The objective function is the total amount of flow out of the source node. The first two sets of constraints are flow balance constraints, ensuring the flow into a given node is equivalent to the flow out of that node. The next two sets of constraints are capacity constraints, ensuring that the amount of flow traveling along an arc or node is at most the capacity of that arc or node. We do not define these constraints for the source and sink nodes, as the remaining flow balance constraints will ensure that the flow out of the source will be the same as into the sink, and the source and sink are uncapacititated. The max flow problem can be equivalently modeled as adding an arc from the sink to the source and finding the maximum circulation through the network, in which case these additional flow balance constraints would be necessary. We note that if $z_{ij} = 0$ for all $(i,j) \in A^R$, then the problem reduces to the standard MFNIP. In Section 3.3, we discuss how to model city-level drug trafficking networks, both in defining the set of arcs and set of restructurable arcs for these networks, and in defining constraints on interdictions and restructurings. Before we do so, we discuss properties of the problem for general networks.

3.2.2 Properties of Problem

We consider how the interdiction decisions can trigger restructuring decisions and how the overall effects of these decisions can impact the max flow in the network. We will show that only certain classes of restructuring decisions can be guaranteed to *not increase* the maximum flow. This will help shed insight into 'bad' disruption strategies for our motivating applications in drug trafficking. We consider the example network in Figure 3.1 that helps to illustrate that implementing available interdiction decisions (even when our budget allows it) may not be optimal and, counter-intuitively, increase the flow in the network. This example is based on modeling city-level drug trafficking. For this example, nodes are labeled with $(flow, capacity)$, and current arcs are uncapacitated. Figure 3.2a demonstrates the maximum flow through this network.

In the example network, we only have the budget to interdict one node in the bottom

tier. The set of restructurable arcs includes arcs that start at the left or right node in the middle tier and end at the capacity 4 nodes in the bottom tier, and arcs that start at the middle node in the middle tier and end at any node it does not currently reach. The defender has the budget to restructure any one arc. We assume that only arcs where one endpoint has an interdicted neighbor can be restructured, as we specifically want to model how the impacted individuals react to the disruptions; in modeling a drug trafficking network spanning multiple regions, the removal of a drug dealer in one region would not impact the operations of a dealer in a different region.

For the MFNIP in this network, as demonstrated in Figure 3.2b, it is optimal to interdict one of the two nodes with a capacity of 4. However, once we incorporate restructuring, interdicting that node allows for the dashed arc in 3.2c to be restructured, allowing for flow to be shifted in a way that increases the total flow through the network. With this set of restructurable arcs, it is optimal to interdict either node with capacity 7, as the arcs that would be allowed to restructure after this interdiction end at nodes already at capacity. Further, note that if we could *only* interdict the nodes with capacity 4, instead of being able to interdict a node in the bottom tier, then it is optimal to not implement any interdiction decisions, as the resulting restructuring will always increase the flow to be more than in the original network.

In this example network, there are some nodes in the bottom tier that have a flow less than their capacity. Our model would not allow arcs to be restructured ending at these nodes, even though one might suspect that, in practice, the nodes would want to. We can think of this as modeling a closed market, where dealers may not want to sell to users they are unfamiliar with, unless the users they currently sell to have been interdicted [100].

We first wish to find cases for interdiction decisions under which the value of the flow will not increase, regardless of restructuring decisions. We do so by considering the auxiliary network $D_f = (N, A_f)$ of the optimal maximum flow in the current network. We construct this auxiliary network by the following procedure for every $(i, j) \in A$. If $x_{ij} < u_{ij}$, then $(i, j) \in A_f$. If $x_{ij} > 0$, then $(j, i) \in A_f$. Define the corresponding minimum (s, t)-cut to be $[U, \bar{U}]$, where a node $i \in U$ if i can be reached from α in D_f, and $i \in \bar{U}$ otherwise. We consider the type of an arc (i, j) that is allowed to be restructured. Proofs of these lemmas appear in Appendix C.

Lemma 3. Restructuring an arc (i, j), with $i \in \bar{U}$ will not increase the value of the flow.

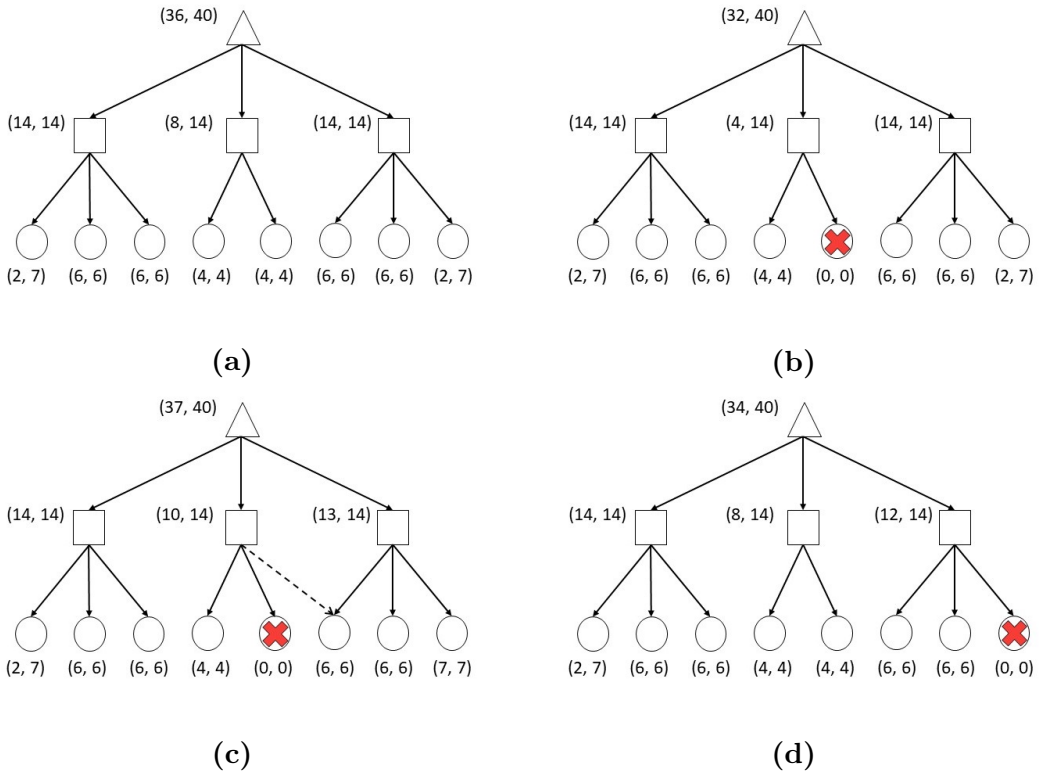


Figure 3.2: How the optimal interdiction plan changes when restructuring is allowed in an example network, where (a) is the optimal flow, (b) is the optimal interdiction before restructuring, (c) is the restructuring in response to the optimal interdiction, and (d) is the optimal interdiction after accounting for restructuring

Lemma 4. Restructuring an arc (i, j), with $i \in U$ and $j \in U$, will not increase the value of the flow.

Lemma 5. Restructuring an arc (i, j), with $i \in U$ and $j \in \bar{U}$ and where there is no path from j to ω in D_f, will not increase the value of the flow.

These results can be combined into more powerful results about sets of nodes allowed to restructure based on the selected interdiction decisions.

Theorem 3. Let I be the set of arcs that are allowed to restructure based on the selected interdiction decisions. If, for all $(i, j) \in I$, (i, j) satisfies Lemma 3, Lemma 4, or Lemma 5, and any arc $(i, j) \in I$ satisfying Lemma 5 does not have another $(j, k) \in I$ satisfying Lemma 3, then the value of the flow will not increase.

We can interpret these results in terms of *restructuring across the minimum cut* for the original maximum flow. In particular, Lemmas 3 and 4 ensure that the restructuring

decisions do not introduce arcs that cross the minimum cut and Lemma 5 ensures that if the restructured arc crosses the minimum cut, it does not enter a node with a path to the sink.

This interpretation has important policy implications for disrupting drug trafficking networks. For city-level drug trafficking networks, Malaviya et al. [75] have shown that the minimum cut in the network tends to be be between the drug dealer and user level. Therefore, our results suggest that for this application, a focus on interdictions at this level will not impact the overall prevalence of drug trafficking. This mathematically confirms the analysis of criminal justice scholars [16].

3.3 Modeling Restructuring in Drug Trafficking Networks

We focus our discussion on modelling hierarchical networks, since the field has found value in viewing drug trafficking networks as hierarchical networks. Each tier is representative of the node's role in the functionality of the network [9, 75]. The first (bottom) tier is representative of drug users, the second tier is representative of drug dealers, the third tier is representative of safe houses of distinct organizations, while the tiers above that are representative of management in distinct organizations [75]. When we refer to a *safe house*, we are referring to the individual(s) responsible for operating it and ensuring its role as a distributor of drugs to dealers. Distinct organizations are constituted from nodes in the safe house tier through the top tier, and there are no arcs that start at a node in one organization and end at a node in a different organization. The nodes in the highest tier for each organization are the suppliers for the organization. Arcs in these networks always start at the i^{th} tier and end at the $(i-1)^{th}$.

3.3.1 Modeling Interdiction Decisions

We first define the set of feasible interdiction decisions Y. The first constraint is one that all interdiction models include, a budget constraint. Suppose c_i is the cost to interdict node i, and a budget b. We define the budget constraint as:

$$\sum_{i \in N} c_i y_i \leq b. \tag{3.4}$$

In disrupting drug trafficking networks, we need to interdict participants in a way that allows us to gather information about the participants above them. Malaviya et al. [75] model this as climbing the ladder constraints. Let τ_i be the number of child nodes of node i

that need to be interdicted before node i can be interdicted. Climbing the ladder constraints can be defined as:

$$\tau_i y_i \leq \sum_{(i,j)\in A} y_j \text{ for all } i \in N. \tag{3.5}$$

It may also be that we want to enforce that some nodes from a set of nodes are required to be interdicted. In the application of drug trafficking, we may want to understand how disrupting the high-level operations of the network triggers restructuring decisions. Let H be the set of nodes we want to interdict at least g nodes from. We can define this constraint as:

$$\sum_{i\in H} y_i \geq g. \tag{3.6}$$

In our application, we want to understand how disrupting the leadership of larger organizations impacts the restructuring of the network. In order to do this, we define H to be the set of nodes in the top two tiers of the two largest organizations. We refer to this as 'interdicting leadership'.

3.3.2 Modeling Restructuring

We now focus on how we model the reactions of the network to interdiction efforts. We first define what arcs belong to the set A^R. Since city-level drug trafficking are modeled as hierarchical networks, we define the base set of restructurable arcs to follow the same rules as arcs currently in the network do. This means that the initial set A^R consists of arcs that could have belonged in the original arc set. These arcs would be representative of participants in the drug trafficking network forming connections with new individuals to sell their product to.

Before defining the constraints on the defender's decisions, we wish to capture which participant is initiating the restructuring efforts, not just the appearance of new arcs. We do so by defining $z = (z_{ij}^{in}, z_{ij}^{out})$, based on whether a node is able to restructure to a tier above or below them, respectively. More formally, z_{ij}^{in} is the indicator of restructuring arc (i, j) when node j is able to restructure, and z_{ij}^{out} is the indicator of restructuring an arc when node i is able to restructure. For applications in disrupting drug trafficking networks, we model that participants in the network may seek to replace connections when either those above them (e.g., a user replacing a dealer) or below them (e.g., a safe house/distributor

replacing a dealer) have been interdicted. Therefore, it is important to note whether the higher level or lower level participant is reaching out to establish the new connection, as we will assume there is a limit to the number of re-connections a participant can establish.

We now describe constraints on the decisions of the defender, $Z(y)$. We assume that they have structured their current network (pre-interdiction) systematically and thus will only allow a small number of replacements to occur if the interdictions impact the participants a node is directly connected to. First, we want to restrict the number of arcs leaving node i. In the drug trafficking network, this indicates that each participant's resources to restructure their network are limited or they desire to only react in a small way to limit the number of new participants that know about them. We only allow l_i new arcs to be created based on actions taken by given node i, signifying that each participant only has time to develop a limited number of new connections below them. We can capture this with the following constraint:

$$\sum_{(i,j)\in A^R} z_{ij}^{out} \leq l_i \text{ for all } i \in N. \tag{3.7}$$

Next, we want to restrict the number of arcs entering node j. In our drug trafficking network example, this could be represented multiple ways. At the lowest level, a drug user does not need a large number of dealers, and may want to limit the number of dealers they purchase from in order to limit the number of risky situations they put themselves in. Similarly for higher levels, participants may want to work with a smaller number of people so that the they can build trust amongst one another and decrease the chances of a connection providing information about them to law enforcement. We only allow s_j new arcs to be created by the actions of a given node j, limiting the number of connections a participant can form with participants above them. We can define the constraint limiting the number of arcs entering a node during the restructuring as:

$$\sum_{(i,j)\in A^R} z_{ij}^{in} \leq s_j \text{ for all } j \in N. \tag{3.8}$$

The next restriction we make is to only allow nodes that are incident to an interdicted arc to replace a disrupted connection. This is to capture that it is likely that local 'reactions' will occur within the network, i.e., if the interdiction did not disrupt the status quo for a certain segment of the network, this segment would have no reason to restructure itself. For example, in order to make up for their lost earnings from losing a drug user, a drug dealer

may seek out new users to sell to. A parameter q_i for node i determines how many new arcs are allowed to be added in and out of node i *per interdicted participant* connected to node i. Intuitively, interdicting a parent node allows for restructuring to a new parent node, and interdicting a child node allows for restructuring to a new child node. We can model this constraint as:

$$\sum_{(i,j)\in A^R} z_{ij}^{out} \leq q_i \sum_{h:(i,h)\in A} y_h \text{ for all } i \in N. \tag{3.9}$$

$$\sum_{(i,j)\in A^R} z_{ij}^{in} \leq q_j \sum_{h:(h,j)\in A} y_h \text{ for all } j \in N. \tag{3.10}$$

Observe that if no child node of a node i has been interdicted, then $y_h = 0$ for all h, $(i,h) \in A$. Thus, the sum on the right side of the inequality (3.9) will be zero. This then forces $z_{ij}^{out} = 0$ for all j, $(i,j) \in A^R$, capturing the desired behavior. The same behavior occurs for parent nodes of a node j.

In order to not have the same arc added twice, we add the following constraint to ensure that both $z_{ij}^{out} = 1$ and $z_{ij}^{in} = 1$ cannot happen.

$$z_{ij}^{in} + z_{ij}^{out} \leq 1 \text{ for all } (i,j) \in A^R. \tag{3.11}$$

The last constraint we have is a budget constraint. Similarly to the attacker having a budget b to interdict the network with, we limit the amount of resources to restructure the entire network, where a_{ij} is the amount of resources needed to restructure arc (i,j), and r is the maximum amount of resources available to restructure the network. This gives us the final constraint:

$$\sum_{(i,j)\in A^R} a_{ij} z_{ij} \leq r. \tag{3.12}$$

With these four sets of constraints, we are able to capture the behavior of the defender reacting to attacker's decisions. We formalize the set $Z(y)$ as:

$$\sum_{(i,j)\in A^R} z_{ij}^{out} \leq l_i \text{ for all } i \in N \tag{3.13}$$

$$\sum_{(i,j)\in A^R} z_{ij}^{in} \leq s_j \text{ for all } j \in N \tag{3.14}$$

$$\sum_{(i,j)\in A^R} z_{ij}^{out} \leq q_i \sum_{h:(i,h)\in A} y_h \text{ for all } i \in N \tag{3.15}$$

$$\sum_{(i,j)\in A^R} z_{ij}^{in} \leq q_j \sum_{h:(h,j)\in A} y_h \text{ for all } j \in N \tag{3.16}$$

$$z_{ij}^{in} + z_{ij}^{out} \leq 1 \text{ for all } (i,j) \in A^R \tag{3.17}$$

$$\sum_{(i,j)\in A^R} a_{ij}(z_{ij}^{in} + z_{ij}^{out}) \leq r \tag{3.18}$$

$$z_{ij}^{in}, z_{ij}^{out} \in \{0,1\} \text{ for all } (i,j) \in A^R \tag{3.19}$$

The above defines our base instances for restructuring in city-level drug trafficking networks. We additionally consider two variants on top of this base set of arcs. We first discuss the recruitment of new participants, then discuss promotion of participants and recruitment across organizations (the two of which we refer to as organizational restructuring). We model the recruitment of new participants as nodes that are not yet connected to the source. For a new participant node j, we add the arc (j,k) to the currently existing arc set, so the new participant can reach the layer below it. However, we add arcs (i,j) to the set of restructurable arcs. For example, recruitable drug users would have an arc to the sink in the original network, and arcs to drug dealers in the restructurable set. Since there are no arcs (i,j) in the network, there is no path from the source node to j, and thus no flow into j. To restructure arc (i,j), we would need to interdict a node h with $(i,h) \in A$, which, intuitively, is interdicting a currently existing child node of i and, therefore, h and j would have similar roles in the network. When an arc (i,j) is added to the network, there is the possibility of completing a path from the source, allowing the possibility of flow through j. This assumes each recruitable participant knows of other participants in the network, such as recruitable dealers knowing of users to sell to. Note that no new constraint needs to be added to include this modeling adaptation, as recruiting a new participant is incorporated in (3.9).

We now seek to model organizational restructuring, which allows for the promotion of participants currently in an organization, as well as the recruitment of participants from one organization into another. We model promotion as including some arcs $(i,j) \in A^R$, where node i is two tiers above j, instead of the typical one tier difference. When including this arc, we want to ensure that our promoted node is replacing one of its parent nodes. This means we only allow z^{in} to take a non-zero value, as z^{in} is allowed to be non-zero when a parent node is interdicted. Let P be the set of promotable nodes. For all $p \in P$, we enforce

$$z_{ip}^{out} = 0 \text{ for all } (i,p) \in A^R. \tag{3.20}$$

We model recruitment from one organization to another as including some arcs $(i,j) \in$

A^R where node i belongs to one organization and node j belongs to a different organization. As per the arcs described in the base instance, node i is only one tier above node j. We want a node j to be able to recruited only when the node they work for has been interdicted; a higher level participant has no incentive to join a different organization unless it is necessary for them to continue their operations. This again translates into only wanting z_{ij}^{in} to be able to be non-zero. Let C be the set of nodes eligible for cross-organizational recruitment. For all $c \in C$ we enforce

$$z_{ic}^{out} = 0 \text{ for all } (i, c) \in A^R. \tag{3.21}$$

We note that, only budget constraints are needed to define Y and $Z(y)$. If $Z(y) = Z$ for all $y \in Y$, meaning our restructuring decisions are independent of interdiction decisions, our method reduces to the same method as [131], which the method of [129] will also reduce to.

3.4 Our Algorithm

We now derive our column-and-constraint (C&CG) algorithm to solve MFNIP-R. In our C&CG algorithm, we iteratively generate interdiction plans and minimum cut values associated with feasible restructuring plans constructed from previously considered restructuring plans, then determine what the optimal restructuring plan is to that interdiction plan. We note that, while our method is derived specifically for the application of disrupting city-level drug trafficking networks, the following derivation can be used for a general network.

In Yue et al. [129], when a previously visited lower level integral solution is infeasible with respect to the current upper level solutions, we exclude the constraints in the upper level problem associated with the lower level solution. We propose a novel projection scheme based on the monotonicity of $Z(y)$ to find a non-trivial lower level decision that satisfies the constraints enforced by the new upper level decision. In other words, we use *partial information* obtained in previous iterations to tighten the bounds on the true objective function of a selected upper level decision. The method we derive is independent of network structure. As such, we will derive our method for a general network. Before proceeding with the description of the algorithm, we first define what is means for $Z(y)$ to be monotonic, as defined in Assumption 1 in [49].

Definition 1. A set $Z(y)$ is **monotonic** if for a fixed $y \in Y$, for all $z \in Z(y)$, if $0 \leq \hat{z} \leq z$, then $\hat{z} \in Z(y)$.

Based on the constraints defined in Section 3.3, $Z(y)$ is monotonic. Intuitively, $\hat{z}$ can be thought of as a partial restructuring plan; choosing to restructure less arcs is still a feasible restructuring plan. If any $z_{ij} = 1$ but $\hat{z}_{ij} = 0$, then $\hat{z}$ is restructuring fewer arcs than z. Under this assumption, any constraints defining $Z(y)$ that are independent of y will be satisfied by $\hat{z}$. As long as we can easily evaluate the dependency of the lower level decisions on the upper level decisions, we can find non-trivial partial information responding to new interdiction plans.

To derive the column-and-constraint generation algorithm, we first consider the maximization problem of the defender as two separate maximization problems, first maximizing over restructuring decisions z, then maximizing over flow decisions x. We can then take the dual of the inner-most problem (the flow problem) to offer an alternative formulation. Since the inner-most maximization problem will only have continuous variables, by strong duality of linear programs, the objective values of the bilevel and trilevel formulation would be equivalent. Let π_i^+ be the dual variable associated with the flow balance constraint for flow into node i, π_i^- be the dual variable associated with the flow balance constraint for flow leaving node i, and θ_i and θ_{ij} be the dual variables associated with the capacity constraint for node i and arc (i, j), respectively. We now present the equivalent dual-based formulation of (3.3):

$$\min_{y} \max_{z} \min_{\pi,\theta} \sum_{i \in N \setminus \{\alpha,\omega\}} u_i(1 - y_i)\theta_i + \sum_{(i,j) \in A} u_{ij}\theta_{ij} + \sum_{(i,j) \in A^R} u_{ij}(z_{ij}^{in} + z_{ij}^{out})\theta_{ij} \tag{3.22a}$$

$$\text{s.t. } \pi_j^+ + \theta_{\alpha j} \geq 1 \text{ for } (\alpha, j) \in A \cup A^R \tag{3.22b}$$

$$\pi_j^+ - \pi_i^- + \theta_{ij} \geq 0 \text{ for } (i, j) \in A \cup A^R \text{ s.t. } i \neq \alpha, j \neq \omega \tag{3.22c}$$

$$\pi_i^- - \pi_i^+ + \theta_i \geq 0 \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.22d}$$

$$-\pi_i^- + \theta_{i\omega} \geq 0 \text{ for } (i, t) \in A \cup A^R \tag{3.22e}$$

$$\theta \geq 0 \tag{3.22f}$$

$$z^{in}, z^{out} \in Z(y) \tag{3.22g}$$

$$y \in Y \tag{3.22h}$$

We note that the dual to the max flow problem is the min cut problem. While there are many solutions to the linear program, we can identify when a solution corresponds to a feasible cut.

Definition 2. We say a solution (π^+, π^-, θ) is a **feasible cut solution** if the following conditions are true:

- If node i is on the source side of the cut, $\pi_i^+ = \pi_i^- = 1$ and $\theta_i = 0$
- If a node i is on the sink side of the cut, $\pi_i^+ = \pi_i^- = 0$ and $\theta_i = 0$
- If a node i is in the cut, $\pi_i^+ = 1$, $\pi_i^- = 0$, and $\theta_i = 1$
- If an arc (i, j) is in the cut, $\theta_{ij} = 1$, and otherwise $\theta_{ij} = 0$

In the classical minimum cut problem, the constraint matrix is totally unimodular, which implies that the continuous variables will be integral in the optimal solution to the linear program. Likewise, for fixed (y, z), the inner-most minimization problem of (3.22) will also satisfy this property. Without loss of generality, any solution (π^+, π^-) can be shifted to fit the above definition. Thus, there must be an optimal solution that is a feasible cut solution.

In a standard network interdiction problem, we would use the McCormick inequalities to linearize the $y_i\theta_i$ terms. However, in the context of our C&CG algorithm, using the McCormick inequalities to linearize the $z_{ij}\theta_{ij}$ terms will result in a large number of variables and constraints being generated. We can instead use the following equivalent formulation of the dual problem, similar to Model 1D presented in [127]:

$$\min_{y \in Y} \max_{z} \min_{\pi, \theta} \sum_{i \in N \setminus \{\alpha, \omega\}} u_i \theta_i + \sum_{(i,j) \in A \cup A^R} u_{ij} \theta_{ij} \tag{3.23a}$$

$$\text{s.t. } \pi_j^+ + \theta_{\alpha j} \geq 1 \text{ for } (\alpha, j) \in A \tag{3.23b}$$

$$\pi_j^+ - \pi_i^- + \theta_{ij} \geq 0 \text{ for } (i, j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega \tag{3.23c}$$

$$\pi_i^- - \pi_i^+ + \theta_i \geq -y_i \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.23d}$$

$$-\pi_i^- + \theta_{i\omega} \geq 0 \text{ for } (i, \omega) \in A \tag{3.23e}$$

$$\pi_j^+ + \theta_{\alpha j} \geq z_{\alpha j}^{in} \text{ for } (\alpha, j) \in A^R \tag{3.23f}$$

$$\pi_j^+ - \pi_i^- + \theta_{ij} \geq z_{ij}^{out} - 1 \text{ for } (i, j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega \tag{3.23g}$$

$$\pi_j^+ - \pi_i^- + \theta_{ij} \geq z_{ij}^{in} - 1 \text{ for } (i, j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega \tag{3.23h}$$

$$\theta \geq 0 \tag{3.23i}$$

$$z^{in}, z^{out} \in Z(y) \tag{3.23j}$$

$$y \in Y \tag{3.23k}$$

We note that in (3.23), we have no z_{sj}^{out}, z_{it}^{in}, z_{it}^{out} variables. This is because the source and sink nodes are not representative of participants in the network, and thus have no

restructuring capabilities of their own, removing the need to define $z^{out}_{\alpha j}$ and $z^{in}_{i\omega}$. We have no $z^{out}_{i\omega}$ because we assume drug dealer nodes cannot restructure to the sink node (i.e., they would be selling their product to themselves), and that recruitable drug users are already connected to the sink, not a drug dealer. We do still have $z^{in}_{\alpha j}$ variables in the case where a node can be promoted to become a supplier.

Similarly to (3.22), for any feasible (y, z), there will be an optimal solution to (3.23) where the continuous variables will be integral. This allows for the following theorem.

Theorem 4. The objective value of the optimal solution to (3.22) is equivalent to that of the objective value of the optimal solution to (3.23).

Proof. Let $y \in Y$ be a feasible interdiction plan and $z \in Z(y)$ be a feasible restructuring plan. We demonstrate that any feasible solution that corresponds to a feasible cut in one program can be converted to a feasible solution that corresponds to a feasible cut in the other.

We first consider a feasible solution (π^+, π^-, θ) in (3.22) that corresponds to a feasible cut solution. We now construct a feasible solution $(\hat{\pi}^+, \hat{\pi}^-, \hat{\theta})$ to (3.23) with the same objective value. We set $\hat{\pi}^+ = \pi^+$ and $\hat{\pi}^+ = \pi^+$, and will construct $\hat{\theta}$ such that the objective value associated with $(\hat{\pi}^+, \hat{\pi}^-, \hat{\theta})$ will be the same as the objective value associated with $(\pi^+, \pi^-\theta)$. With these choices of $\hat{\pi}^+$ and $\hat{\pi}^-$, then $\hat{\theta}_{ij} = \theta_{ij}$ for all $(i, j) \in A$ will be feasible. Note that if $\theta_i = 0$ for $i \in N$, then $\pi^+_i \leq \pi^-_i$. In (3.22), we will have that $(1 - y_i)\theta_i = 0$ regardless of y_i. Likewise, in (3.23), the corresponding constraint will evaluate to $\theta_i \geq -y_i$. Since $y_i \in \{0, 1\}$, then $-y_i \leq 0$. Thus, $\theta_i = 0$ is feasible in (3.23).

Now consider where $\theta_i = 1$ in (3.22). Note that if $\theta_i = 1$, it must be that $\pi^-_i - \pi^+_i = -1$, else $\theta_i = 0$ will be feasible and the solution is not a feasible cut solution. In (3.23), the corresponding constraint will evaluate to $-1 + \theta_i \geq -y_i$. We have two cases: $y_i = 0$ and $y_i = 1$. If $y_i = 0$, then $(1 - y_i)\theta_{ij} = 1$. In (3.23) the corresponding constraint further evaluates to $-1 + \theta_i \geq 0$, thus enforcing $\theta_i \geq 1$. Thus, we will have that $\hat{\theta}_i = 1 = (1 - y_i)\theta_i$ is feasible. If $y_i = 1$, then $(1 - y_i)\theta_i = 0$. In (3.23), the corresponding constraint evaluates to $-1 + \theta_i \geq -1$. Thus, $\theta_i \geq 0$ is feasible, and we will have that $\hat{\theta}_i = 0 = (1 - y_i)\theta_i$. Thus, $\hat{\theta}_i$ in (3.23) will take the same value as $(1 - y_{ij})\theta_{ij}$ in (3.22), resulting in the same objective values. Similar arguments can be used to construct feasible $\hat{\theta}_{ij}$ for $(i, j) \in A^R$ such that $z_{ij}\theta_{ij} = \hat{\theta}_{ij}$ for all $(i, j) \in A^R$. Since this can be done for any feasible (y, z), any feasible solution in (3.22) that corresponds to a feasible cut solution has an equivalent solution in (3.23).

To convert from a solution of (3.23) to (3.22), we can follow the process in reverse. Let $(\hat{\pi}^+, \hat{\pi}_-, \hat{\theta})$ be a feasible solution to (3.23) that corresponds to a feasible cut solution. We again take $\pi^+ = \hat{\pi}^+$, $\pi^- = \hat{\pi}^-$, and $\theta_{ij} = \hat{\theta_{ij}}$ for $(i,j) \in A$. If $\hat{\theta}_i = 1$, we also have that $\theta_i = 1$ is feasible. Suppose $\hat{\theta}_i = 0$. We have two cases, $y_i = 0$ and $y_i = 1$. If $y_i = 0$, then $\hat{\pi}_i^- \geq \hat{\pi}_i^+$, and thus $\theta_i = 0$ is feasible. This choice results in $(1 - y_i)\theta_i = 0 = \hat{\theta}_i$. If $y_i = 1$, we again have two cases. If $\hat{\pi}_i^- \geq \hat{\pi}_i^+$, then $\theta_i = 0$ is feasible. Otherwise, if $\hat{\pi}_i^- - \hat{\pi}_i^+ = -1$, we must have $\theta_i \geq 1$ to maintain feasibility in (3.22). However, with $y_i = 1$, $(1 - y_i)\theta_i = 0 = \hat{\theta}_i$. As before, similar arguments can be used to construct feasible θ_{ij} for $(i,j) \in A^R$ such that $z_{ij}\theta_{ij} = \hat{\theta}_{ij}$. Since this can be done for any feasible (y,z), any feasible cut solution to (3.23) that corresponds to a cut solution has an equivalent solution in (3.22).

For fixed (y,z), we know that an optimal solution of the inner-most minimization problem will correspond to a feasible cut solution. Let $f^*_{(2)}$ be the optimal objective value of the inner-most minimization problem of (3.22) and $f^*_{(3)}$ be the optimal objective value of the inner-most optimization problem of (3.23). Consider the optimal solution $(\pi^{+^*}, \pi^{-^*}\theta^*)$ of (3.22) that is a feasible cut solution, with objective value $f^*_{(2)}$. We can convert this solution to a feasible cut solution of (3.23) $(\hat{\pi}^{+^{(2)}}, \hat{\pi}^{-^{(2)}}, \hat{\theta}^{(2)})$, with equivalent objective value $f^*_{(2)}$. Since this solution is feasible in (3.23), we have that $f^*_{(3)} \leq f^*_{(2)}$. Likewise, consider the optimal solution $(\hat{\pi}^{+^*}, \hat{\pi}^{-^*}, \hat{\theta}^*)$ of (3.23) that is a feasible cut solution, with objective value $f^*_{(3)}$. We can also convert this solution to a feasible cut solution of (3.22) $(\pi^{+^{(3)}}, \pi^{-^{(3)}}, \theta^{(3)})$, with equivalent objective value $f^*_{(3)}$. Since this solution is feasible in (3.22), we have that $f^*_{(2)} \leq f^*_{(3)}$. Thus, for fixed (y,z), we have that $f^*_{(2)} = f^*_{(3)}$. Since this holds regardless of choice of (y,z), it must be that, for any (y,z), the inner-most minimization problems of the two programs will have equivalent objective values. Thus, the objective values of (3.22) and (3.23) will be equivalent. □

The difference between the two models is quite subtle. Where (3.22) computes the cut in the original network, then determines that interdicted arcs should not be counted towards the value and restructured arcs should be counted towards the value of the cut (if they are in it), (3.23) determines exactly what the cut is in the network after interdictions and restructuring have occurred.

We now remark that there are a finite number of interdiction plans, so Y is finite. Additionally, for a given interdiction plan y, $Z(y)$ is finite. Thus, $\bigcup_{y \in Y} Z(y)$ is finite. For a given $y \in Y$, we can replace the maximization problem by enumerating over all possible

decisions $z \in Z(y)$, defining π^z and θ^z to be the partition of the node set and indicator of which arcs are in the cut corresponding to the network $G(z) = (N, A \cup \{(i,j) \in A^R : z_{ij} = 1\})$. We additionally make the objective value associated with each z a constraint, and enforce that η, the objective value of the bilevel problem, is at least as large as the objective value associated with each z. The following formulation is non-standard, since the set of constraints depends on the decision y.

$$\min_{y,\pi,\theta,\eta} \eta \tag{3.24a}$$

$$\text{s.t. } \eta \geq \sum_{i \in N \setminus \{\alpha,\omega\}} u_i \theta_i^z + \sum_{(i,j) \in A \cup A^R} u_{ij} \theta_{ij}^z \text{ for } z \in Z(y) \tag{3.24b}$$

$$\pi_j^{+^z} + \theta_{\alpha j}^z \geq 1 \text{ for } (\alpha, j) \in A, z \in Z(y) \tag{3.24c}$$

$$\pi_j^{+^z} - \pi_i^{-^z} + \theta_{ij}^z \geq 0 \text{ for } (i,j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega, z \in Z(y) \tag{3.24d}$$

$$\pi_i^{-^z} - \pi_i^{+^z} + \theta_i^z \geq -y_i \text{ for } i \in N \setminus \{\alpha, \omega\}, z \in Z(y) \tag{3.24e}$$

$$-\pi_i^{-^z} + \theta_{i\omega}^z \geq 0 \text{ for } (i,\omega) \in A, z \in Z(y) \tag{3.24f}$$

$$\pi_j^{+^z} + \theta_{\alpha j}^z \geq z_{\alpha j}^{in} \text{ for } (\alpha, j) \in A^R, z \in Z(y) \tag{3.24g}$$

$$\pi_j^{+^z} - \pi_i^{-^z} + \theta_{ij}^z \geq z_{ij}^{out} - 1 \text{ for } (i,j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega, z \in Z(y) \tag{3.24h}$$

$$\pi_j^{+^z} - \pi_i^{-^z} + \theta_{ij}^z \geq z_{ij}^{in} - 1 \text{ for } (i,j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega, z \in Z(y) \tag{3.24i}$$

$$\theta \geq 0 \tag{3.24j}$$

$$y \in Y \tag{3.24k}$$

In (3.24), the restructuring decisions z now become parameters, as we enumerate over all possible restructuring decisions, leaving the attacker as the only decision-maker. In terms of our problem, the method of Yue et al. [129] would convert this non-standard formulation to a standard formulation by including implication constraints to determine if a restructuring plan is feasible with respect to the current interdiction plan, and only including constraints associated with feasible restructuring plans. These implication constraints prevent infeasible solutions from being considered when determining a lower bound on the objective function, which would invalidate the lower bound. They then reduce the size of the problem by only considering a subset of restructuring plans, then iteratively adding new plans that are relevant to determining the true objective value. However, if making a small change to the interdiction plan results in making many of the restructuring plans infeasible, then the lower bound on the true objective value will grow at a slower rate, causing the algorithm to converge slowly.

Instead of removing constraints associated with infeasible restructuring plans, we ex-

ploit the monotonicity of z to identify the components of each restructuring plan that are feasible after changing the interdiction decision. Whenever we can identify a non-trivial restructuring plan from an infeasible restructuring plan, we include an additional valid lower bound on the true objective value that the method of Yue et al. [129] would exclude. Figure 3.3 demonstrates the benefit of this inclusion.

Consider the network in Figure 3.3a. Suppose the attacker has the budget to interdict at most two nodes in the bottom tier, any interdiction allows the parent node of an interdicted node to restructure to any one node in the bottom tier, and the defender can restructure at most two arcs. Suppose we know the interdiction plan and restructuring plan identified in Figure 3.3b. If we were to change our interdiction plan to the plan identified in Figure 3.3c, the dotted arc on the right is not allowed to be restructured, but the dashed arc on the left is still able to be restructured. The method of [129] would determine that, since one arc cannot be restructured, the entire restructuring plan must be discarded, as demonstrated in Figure 3.3c. This method would tell us that the objective value associated with that interdiction plan would be at least 31. However, if we instead consider the restructuring sub plan of just the dashed arc on the left, as demonstrated in Figure 3.3d, we can observe that the objective value associated with that interdiction plan is higher than what the method of [129] would identify. Our method would identify that the objective value associated with that restructuring plan would be at least 34. Because of this, applying their method will require more iterations to identify the optimal solution than our method will, since their method results in a slower increase in the lower bound on objective value.

We now demonstrate how to identify feasible restructuring sub plans. Consider two interdiction plans $\bar{y}, \hat{y} \in Y$ where $\hat{y} \neq \bar{y}$. For a restructuring plan $\bar{z} \in Z(\bar{y}) \setminus Z(\hat{y})$ we wish to identify some point $\hat{z} \in Z(\hat{y})$ that is "near" $\bar{z}$, i.e., we want to identify the restructuring plan that restructures as many arcs in $\bar{z}$ as possible while still being feasible with respect to the new interdiction plan $\hat{y}$. We define $w = (w^{in}, w^{out})$ to be the indicator of which arcs restructured in $\bar{z}$ are still able to restructured in response to the new interdiction plan $\hat{y}$. Referencing the network in Figure 3.3, if $\bar{y}$ was the interdiction plan in Figure 3.3b and $\bar{z}$ was the restructuring plan in Figure 3.3b, then we would want to define constraints on y and w such that when we consider $\hat{y}$ to be the interdiction plan in Figure 3.3d, we are able to use the w variables to identify the restructuring plan in Figure 3.3d as $\hat{z}$.

Since we have variables z^{in} and z^{out}, we need to have variables w^{in} and w^{out} in our

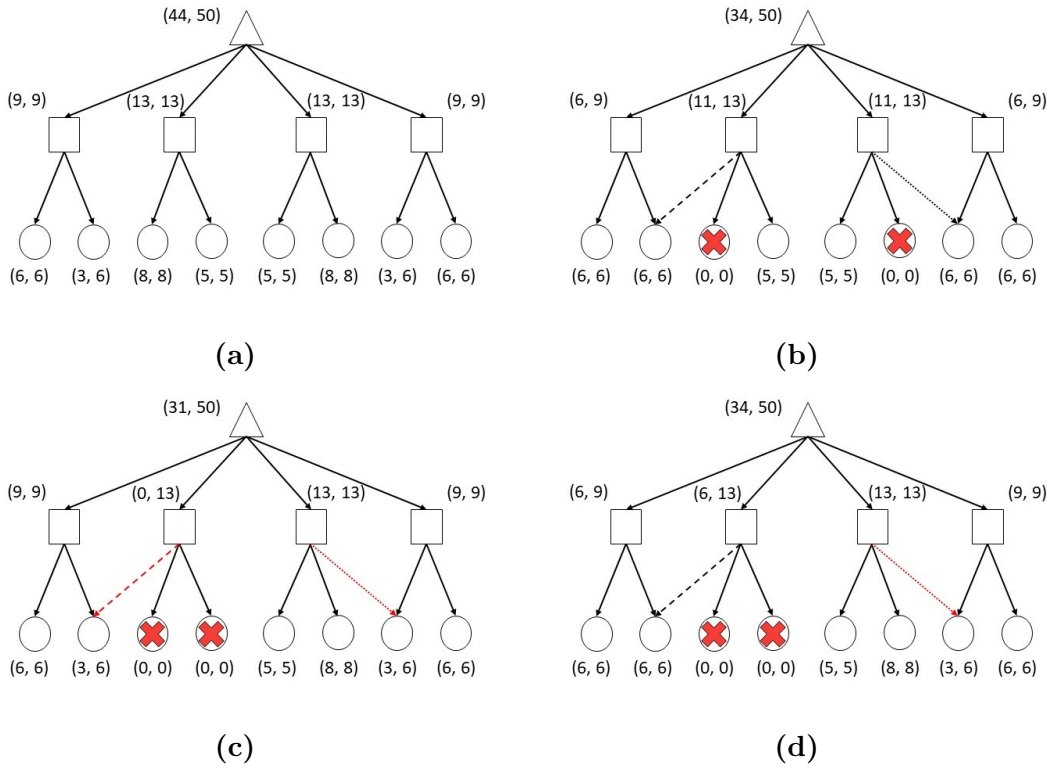


Figure 3.3: Impact of identifying restructuring sub plans on determining the objective value of the MFNIP-R where (a) is the optimal flow, (b) is a sample interdiction and restructuring plan, (c) is the restructuring plan without partial information, and (d) is the restructuring plan with partial information

model. Because of this, we also need to adapt our model to reflect having two ways to restructure. Our constraints involving restructuring decisions are of the form $\pi_j^z - \pi_i^z + \theta_{ij}^z \geq z_{ij} - 1$. When $w_{ij} = 1$, this does not consider whether $z_{ij} = 1$ because $z_{ij}^{out} = 1$ or $z_{ij}^{in} = 1$. This consideration is needed in order to maintain feasibility with respect to the interdiction decisions y. If, for example, $z_{ij}^{in} = 1$, but $w_{ij}^{in} = 0$ and $w_{ij}^{out} = 1$, we would not have that (i, j) is a restructured arc in the sub plan, as no interdiction has occurred to allow this restructuring. To incorporate this, we need to replace the above set of constraints with a pair of constraints, $\pi_j^k - \pi_i^k + \theta_{ij}^k \geq z_{ij}^{out,k} + w_{ij}^{out,k} - 2$ and $\pi_j^k - \pi_i^k + \theta_{ij}^k \geq z_{ij}^{in,k} + w_{ij}^{in,k} - 2$. With this pair of constraints, the relationship between π_i, π_j and θ_{ij} is only enforced when $z_{ij}^{out,k} = 1$ and $w_{ij}^{out,k} = 1$, or when $z_{ij}^{in,k} = 1$ and $w_{ij}^{in,k} = 1$.

We now derive constraints based on equations (3.7) - (3.12) from Section 3.3. By monotonicity, any sub plan of a previously visited restructuring plan will satisfy constraints (3.7), (3.8), (3.11), and (3.12), since there is no dependency on y in these constraints. What

remains to enforce is feasibility based on constraints (3.9) and (3.10). This means that we need define constraints to determine how many restructurings each node can perform based on a given interdiction plan, and which of the arcs chosen to be restructured in previously considered restructuring plans are still able to be restructured. However, we must also account for the fact that more interdictions can be done than potential allowable restructurings. To account for this, as well as the potential fractionality based on q_i, we define additional variables, $\delta_i^{in,k}, \delta_i^{out,k} \in \{0,1\}$, $\kappa_i^{in,k}, \kappa_i^{out,k} \in \mathbb{Z}_{\geq 0}$, and an additional parameter $\gamma > 0$. The variables $\kappa_i^{in,k}$ and $\kappa_i^{out,k}$ are representative of the number of arcs allowed to be restructured into and out of node i, respectively. The variables $\delta_i^{in,k}$ and $\delta_i^{out,k}$ are indicators of if, for each node i, every arc into or out of i in the k^{th} plan will be restructured, respectively. For each k, we can compute $w^{in,k}$ and $w^{out,k}$ with the following sets of constraints:

$$q_j \sum_{h \in N:(h,j) \in A} y_h - 1 + \gamma \leq \kappa_j^{in,k} \leq q_j \sum_{h \in N:(h,j) \in A} y_h \text{ for } j \in N \setminus \{s,t\} \tag{3.25}$$

$$q_j |\{h \in N : (h,j) \in A\}| \cdot \delta_i^{in,k} + \sum_{i \in N:(i,j) \in A^R, z_{ij}^{in,k}=1} w_{ij}^{in,k} \geq \kappa_j^{in,k} \text{ for } j \in N \setminus \{s,t\} \tag{3.26}$$

$$\delta_j^{in,k} \leq \frac{\sum_{j \in N:(i,j) \in A^R, z_{ij}^{in,k}=1} w_{ij}^{in,k}}{|\{i \in N : (i,j) \in A^R, z_{ij}^{in,k} = 1\}|} \text{ for } j \in N \setminus \{s,t\} \tag{3.27}$$

$$q_i \sum_{h \in N:(i,h) \in A} y_h - 1 + \gamma \leq \kappa_i^{out,k} \leq q_i \sum_{h \in N:(i,h) \in A} y_h \text{ for } i \in N \setminus \{s,t\} \tag{3.28}$$

$$q_i |\{h \in N : (i,h) \in A\}| \cdot \delta_i^{out,k} + \sum_{j \in N:(i,j) \in A^R, z_{ij}^{out,k}=1} w_{ij}^{out,k} \geq \kappa_i^{out,k} \text{ for } i \in N \setminus \{s,t\} \tag{3.29}$$

$$\delta_i^{out,k} \leq \frac{\sum_{j \in N:(i,j) \in A^R, z_{ij}^{out,k}=1} w_{ij}^{out,k}}{|\{j \in N : (i,j) \in A^R, z_{ij}^{out,k} = 1\}|} \text{ for } i \in N \setminus \{s,t\} \tag{3.30}$$

Constraints (3.16) and (3.19) determine the number of arcs that are allowed to be restructured into or out of each node j, capturing the value of $\kappa_j^{in,k} = \lfloor q_j \sum_{h \in N:(h,j) \in A} y_h \rfloor$ and $\kappa_j^{out,k} = \lfloor q_j \sum_{h \in N:(j,h) \in A} y_h \rfloor$, respectively. Note that these sums in the inequalities are the same as the sums in (3.9) and (3.10). Knowing how many arcs are allowed to restructure into or out of each node, constraints (3.17) and (3.20) determine which of the arcs were restructured in the k^{th} restructuring plan should still be restructured, as determined by the attacker. However, we need to prevent infeasibility when the number of arcs that *can be* restructured from a given node is more than the number of arcs that *were* restructured from that node. If the number of arcs that can be restructured into (or out of) a node j is greater than the number of arcs that have been restructured into (or out of) that node in the k^{th}

plan, $\delta_j^{in,k} = 1$ (or $\delta_j^{out,k} = 1$, respectively), allowing the constraint to maintain feasibility. Constraints (3.18) and (3.21) enforce that these indicators are 0 unless every previously restructured arc into (or out of) each node has been determined to also be restructured in a new plan. If q_j is integral, the constraints (3.16) and (3.19) are unnecessary, since the inequalities in the constraint will never be able to take fractional values. In this case, $q_j \sum_{l:(l,j)\in A} y_l$ can be directly substituted in for $\kappa_j^{in,k}$ and $\kappa_j^{out,k}$.

With the inclusion of these constraints, we can now convert (3.24) to a standard formulation, where we enumerate over the fixed set $\bigcup_{y\in Y} Z(y)$ instead of just $Z(y)$. Let $\left|\bigcup_{y\in Y} Z(y)\right| = M$. The following is a standard formulation of (3.24):

$$\min_{y,\pi,\theta,\eta} \quad \eta \tag{3.31a}$$

$$\text{s.t. } \eta \geq \sum_{i\in N\setminus\{\alpha,\omega\}} u_i\theta_i^k + \sum_{(i,j)\in A\cup A^R} u_{ij}\theta_{ij}^k \text{ for } k = 1,\ldots,M \tag{3.31b}$$

$$\pi_j^{+^k} + \theta_{\alpha j}^k \geq 1 \text{ for } (\alpha,j) \in A, k = 1,\ldots,M \tag{3.31c}$$

$$\pi_j^{+^k} - \pi_i^{-^k} + \theta_{ij}^k \geq 0 \text{ for } (i,j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,M \tag{3.31d}$$

$$\pi_i^{-^k} - \pi_i^{+^k} + \theta_i^k \geq -y_i \text{ for } i \in N\setminus\{\alpha,\omega\}, k = 1,\ldots,M \tag{3.31e}$$

$$-\pi_i^{-^k} + \theta_{i\omega}^k \geq 0 \text{ for } (i,\omega) \in A, k = 1,\ldots,M \tag{3.31f}$$

$$\pi_j^{+^k} + \theta_{\alpha j}^k \geq z_{\alpha j}^{in,k} + w_{\alpha j}^{in,k} - 1 \text{ for } (\alpha,j) \in A^R, k = 1,\ldots,M \tag{3.31g}$$

$$\pi_j^{+^k} - \pi_i^{-^k} + \theta_{ij}^k \geq z_{ij}^{out,k} + w_{ij}^{out,k} - 2 \text{ for } (i,j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,M \tag{3.31h}$$

$$\pi_j^{+^k} - \pi_i^{-^k} + \theta_{ij}^k \geq z_{ij}^{in,k} + w_{ij}^{in,k} - 2 \text{ for } (i,j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,M \tag{3.31i}$$

$$\text{Constraints (3.16)} - \text{(3.21) for } i \in N\setminus\{\alpha,\omega\}, k = 1,\ldots,M \tag{3.31j}$$

$$w^k \in \{0,1\}^{|A^R|} \text{ for } k = 1,\ldots,M \tag{3.31k}$$

$$\theta \geq 0 \tag{3.31l}$$

$$y \in Y \tag{3.31m}$$

Now that we have a standard formulation enumerating over fixed set $\bigcup_{y\in Y} Z(y)$, we instead seek to identify a valid lower bound by enumerating over a subset of point in this set. Let $\{z^1,\ldots,z^n\} \subseteq \bigcup_{y\in Y} Z(y)$, where $n \leq M$. By enumerating over these points, we can formulate a program that provides a valid lower bound on (3.31).

$$\min_{y,\pi,\theta,\eta} \eta \tag{3.32a}$$

$$\text{s.t. } \eta \geq \sum_{i\in N\setminus\{\alpha,\omega\}} u_i\theta_i^k + \sum_{(i,j)\in A\cup A^R} u_{ij}\theta_{ij}^k \text{ for } k = 1,\ldots,n \tag{3.32b}$$

$$\pi_j^{+^k} + \theta_{\alpha j}^k \geq 1 \text{ for } (\alpha,j) \in A, k = 1,\ldots,n \tag{3.32c}$$

$$\pi_j^{+^k} - \pi_i^{-^k} + \theta_{ij}^k \geq 0 \text{ for } (i,j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,n \tag{3.32d}$$

$$\pi_i^{-^k} - \pi_i^{+^k} + \theta_i^k \geq -y_i \text{ for } i \in N \setminus \{\alpha, \omega\}, k = 1, \ldots, n \tag{3.32e}$$

$$-\pi_i^{-^k} + \theta_{i\omega}^k \geq 0 \text{ for } (i, \omega) \in A, k = 1, \ldots, n \tag{3.32f}$$

$$\pi_j^{+^k} + \theta_{\alpha j}^k \geq z_{\alpha j}^{in,k} + w_{\alpha j}^{in,k} - 1 \text{ for } (\alpha, j) \in A^R, k = 1, \ldots, n \tag{3.32g}$$

$$\pi_j^{+^k} - \pi_i^{-^k} + \theta_{ij}^k \geq z_{ij}^{out,k} + w_{ij}^{out,k} - 2 \text{ for } (i, j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1, \ldots, n \tag{3.32h}$$

$$\pi_j^{+^k} - \pi_i^{-^k} + \theta_{ij}^k \geq z_{ij}^{in,k} + w_{ij}^{in,k} - 2 \text{ for } (i, j) \in A^R \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1, \ldots, n \tag{3.32i}$$

$$\text{Constraints (3.16)} - \text{(3.21) for } i \in N \setminus \{\alpha, \omega\}, k = 1, \ldots, n \tag{3.32j}$$

$$w^k \in \{0, 1\}^{|A^R|} \text{ for } k = 1, \ldots, n \tag{3.32k}$$

$$\theta \geq 0 \tag{3.32l}$$

$$y \in Y \tag{3.32m}$$

Theorem 5. The objective value of the optimal solution to (3.32) is a lower bound on the objective value of (3.31).

Proof. Let $y \in Y$ and $\{z^1, \ldots, z^n\} \subseteq \bigcup_{y \in Y} Z(y)$, and consider a feasible solution to (3.31), $(\hat{\eta}, \hat{y}, \hat{\pi}^+, \hat{\pi}^-, \hat{\theta})$. Since $\{z^1, \ldots, z^n\} \subseteq \bigcup_{y \in Y} Z(y)$, there are variables $(\hat{\pi}^{+^{z^1}}, \ldots, \hat{\pi}^{+^{z^n}}, \hat{\pi}^{-^{z^1}}, \ldots, \hat{\pi}^{-^{z^n}}, \hat{\theta}^{z^1}, \ldots, \hat{\theta}^{z^n})$ associated with $\{z^1, \ldots, z^n\}$. The solution $(\hat{\eta}, \hat{y}, \hat{\pi}^{+^1}, \ldots, \hat{\pi}^{+^n}, \hat{\pi}^{-^1}, \ldots, \hat{\pi}^{-^n}, \hat{\theta}^1, \ldots, \hat{\theta}^n)$ is a feasible solution in (3.32). Since this solution is feasible in (3.32), we know $\eta^* \leq \hat{\eta}$. Thus, a feasible solution to (3.32) can be created from any solution to (3.31). Thus, the objective value of (3.32) is a lower bound on the objective value of (3.31). □

Once we have solved (3.32) for optimal first stage decision $\hat{y}$, we do not yet know if any z^k is the optimal response to $\hat{y}$. We can easily compute the maximization decisions $\hat{z}$ that are the optimal response to $\hat{y}$. We do so by solving the following:

$$\max_{x,z} \sum_{(\alpha,j) \in A \cup A^R} x_{\alpha j} \tag{3.33a}$$

$$\text{s.t.} \sum_{(j,i) \in A \cup A^R} x_{ji} - x_i = 0 \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.33b}$$

$$x_i - \sum_{(i,j) \in A \cup A^R} x_{ij} = 0 \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.33c}$$

$$0 \leq x_{ij} \leq u_{ij} \text{ for } (i, j) \in A \tag{3.33d}$$

$$0 \leq x_i \leq u_i(1 - \hat{y}_i) \text{ for } i \in N \setminus \{\alpha, \omega\} \tag{3.33e}$$

$$0 \leq x_{ij} \leq u_{ij} z_{ij} \text{ for } (i, j) \in A^R \tag{3.33f}$$

$$z \in Z(\hat{y}) \tag{3.33g}$$

When we solve this problem, we recover a bilevel feasible solution. Thus, the optimal objective value of this program is an upper bound on the true solution. Further, we can

introduce its optimal solution $\hat{z}$ as a new point in the partial enumeration. Thus, we can iteratively find upper bounds U^k and lower bounds L^k on the true objective value of (3.3). We stop the algorithm when $U^k - L^k \leq \epsilon$, where $\epsilon \geq 0$. If we choose $\epsilon = 0$, we solve the problem exactly. We now introduce the column and constraint generation procedure for MFNIP-R as Algorithm 6.

Algorithm 6 C&CG for MFNIP-R

Initialize: lower bound $L = -\infty$, upper bound $U = \infty$, optimal interdiction decisions $y^* = 0$, optimal restructuring decisions $z^1 = z^* = 0$, error tolerance $\epsilon \geq 0$. iteration counter $n = 1$.
while $U - L > \epsilon$ **do**
Step 1. Solve (3.32) for optimal interdiction decision y^n and objective value η^n. Set $L = \eta^n$.
if $U - L \leq \epsilon$ **then**
terminate; y^* and z^* are the optimal decisions with objective value U.
end if
Step 2. Input y^n into (3.33) as data and solve (3.33) for optimal restructuring decisions z^{n+1} and objective value $\sum_{(\alpha,j)\in A\cup A^R} x_{\alpha j}$.
if $x_{ts} < U$ **then**
Let $y^* = y^n$, $z^* = z^{n+1}$, $U = x_{ts}$.
end if
Step 3. Include constraints corresponding to z^{n+1} in (3.32), create variables π^{n+1}, θ^{n+1}, w^{n+1}, set $n = n + 1$, return to Step 1.
end while

Theorem 6. Algorithm 6 converges to the optimal solution of (3.24) when $\epsilon = 0$.

Proof. First, observe that if $\epsilon = 0$, we need $L \geq U$ for Algorithm 6 to terminate, meaning we need $\eta^n \geq \min\{\hat{\eta}^1, \ldots, \hat{\eta}^{n-1}\}$. Suppose that a solution $\hat{y}$ that is optimal in (3.32) is visited twice, at iterations i and j, where $i < j$. The algorithm will terminate, since, by the first set of constraints in (3.32), $\eta^j = \hat{\eta}^i \geq \min\{\hat{\eta}^1, \ldots, \hat{\eta}^{n-1}\}$. Let y^* be the optimal solution to the full enumeration problem with objective value η^*, and suppose that Algorithm 6 never visits y^*. First, consider the case where the algorithm terminates with a solution $\hat{y}$ such that $\hat{\eta} < \eta^*$. Since the algorithm terminated, we found a bilevel feasible solution $(\hat{y}, \hat{z})$ with objective value $\hat{\eta}$. Thus y^* is not the optimal solution, creating a contradiction. Now consider the case where the algorithm terminates with a solution with objective value $\hat{\eta} > \eta^*$. Then there must be some iteration k such that $\eta^k > \eta^*$. By Theorem 5, we can construct a corresponding solution to (3.32) from the optimal solution to (3.31). Since this solution

is feasible in (3.32), $\eta^* \geq \eta^k$, creating a contradiction. Thus, the optimal solution must be visited by Algorithm (6). Since the sets Y and $\bigcup_{y \in Y} Z(y)$ are finite sets, we know that the optimal solution will be reached in finite time. □

From Step 3 of Algorithm 6, we can revisit why we choose to derive the problem from (3.23) instead of (3.22). If we were to use (3.22), we would have bilinearities in $y_i \theta_i^k$ for $i \in N$ for every iteration k. Additionally, including $w^{in,k}$ and $w^{out,k}$ in this model would result in more bilinear terms $z_{ij}^{in,k} w_{ij}^{in,k} \theta_{ij}^k$ and $z_{ij}^{out,k} w_{ij}^{out,k} \theta_{ij}^k$ for all $(i,j) \in A^R$ for every iteration k. Each of these bilinearities would need to be linearized with the McCormick inequalities, resulting in $|N| + 2|A^R|$ new variables and $O(|N| + 2|A^R|)$ new constraints each iteration. These additional variables and constraints can cause the solve time of each iteration to take considerably longer due to the larger problem size.

We note that our idea for utilizing partial information can be modified to fit into other general BIP solvers. The main idea is understanding how previously visited lower level solutions impact the objective value while exploring upper level solutions. As long as we can define linear constraints to determine what integer components of a lower level solution remain feasible, and appropriately adjust the continuous components to maintain feasibility, when changing the upper level solution, we can identify non-trivial feasible lower level solutions that provide a better lower bound on the true objective value. Upon reaching a bilevel feasible solution (y, z), we can add new constraints to the problem determining what a non-trivial sub plan $\hat{z}$ with respect to other upper level decisions $\hat{y}$, updating each non-fathomed node in the branch-and-bound tree. Doing so will allow these solvers to better account for the impact of the lower level problem on the objective value of the BIP as the branching procedure occurs.

3.4.1 Special Case for Constraints on w

In the case where the maximum number of restructuring arcs leaving and entering i is one (i.e., $l_i = s_i = 1$) for all $i \in N$, we can define a different set of constraints to capture restructuring. Instead of defining w^z based on when an arc can appear in each given restructuring plan, we can define w_{ij} as the indicator of when an arc (i,j) can be restructured based on the current interdiction plan. This means that when a given node i is interdicted, every arc (u,v) that can be restructured due to this interdiction will have $w_{uv}^{in} = 1$ or $w_{uv}^{out} = 1$ as appropriate. Since, for each previous plan, at most one of each arcs into (and out of)

each node will have been restructured, at most one arc into (and out of) each node will have both $z_{uv}^{in,k} = 1$ and $w_{uv}^{in} = 1$ (or $z_{uv}^{out,k} = w_{uv}^{out} = 1$, respectively), guaranteeing feasibility. This eliminates the need for δ. Additionally, since we can only restructure one arc into and out of each node, we no longer need to accurately capture how many arcs can be restructured for each node, but instead whether the node as the ability to restructure or not. This allows us to also eliminate κ. Suppose there exist integers a_i, b_i such that $q_i = \frac{a_i}{b_i}$ for each node i. We can instead define our constraints on w^{in} and w^{out} as the following:

$$a_i|\{l : (l,j) \in A\}|w_{ij}^{in} \geq a_i \sum_{l:(l,j)\in A} y_l - b_i + 1 \text{ for } (i,j) \in A^R \tag{3.34}$$

$$a_i|\{l : (l,j) \in A\}|w_{ij}^{out} \geq a_i \sum_{l:(l,j)\in A} y_l - b_i + 1 \text{ for } (i,j) \in A^R \tag{3.35}$$

Note that, if $q_i \geq 1$, we could replace a_i and b_i with 1, as only one interdiction is needed to allow for the one allowed restructuring. Otherwise, when $q_i < 1$, w_{ij}^{in} will only be 1 when $a_i \sum_{l:(l,j)\in A} y_l \geq b_i$, which only occurs when $q_i \sum_{l:(l,j)\in A} y_l \geq 1$. This formulation allows us to generate all needed w variables upfront, as opposed to generating more with each newly identified restructuring plan. In this case, constraints (3.25) and (3.26) would be inserted into (3.31) in place of constraints (3.16) - (3.21).

3.5 Computational Results: Overview and Comparison to Other C&CG Methods

We test our algorithm on five generated networks modeling city-level drug trafficking networks with 200 users, over 10 different attacker budgets and a fixed defender budget. These networks are generated following the procedure of Malaviya et al. [75]. The minimum budget tested allows for interdicting at least one node in the second highest tier for four of the five data sets, which is satisfied by the second smallest budget. These networks were generated following the procedure outlined in [75], each consisting of six tiers. Restructurable arcs between tiers are created by generating the lists of arcs between each pair of sequential tiers, only using nodes currently in the network, then randomly selecting from each list. This guarantees that each pair of sequential tiers has restructurable arcs between them. This constitutes our base data sets. From these base sets, we provide two adaptations. The first adaptation is the inclusion of recruitable participants, members who are not yet in the

network but can be brought in to the network. The number of "recruitable" users, dealers and safe houses is 20% of the number of existing users, dealers and safe houses, respectively. These nodes are connected to the lower tiers of the network similar to [75]. All "recruitable" users and dealers are randomly connected to nodes in the tier above them. "Recruitable" safe house nodes are assigned arcs and restructurable arcs based on an existing safe house, to act as a backup should the existing safe house be interdicted. These generation methods also guarantee that each of these "recruitable" participants has a path from the source through arc restructuring. The second adaptation we call organizational restructuring. This allows for the restructuring of arcs that skip a tier, emulating the promotion of a participant to a higher-level within the network. It also includes the recruitment of participants from other organizations. For our experiments, we set $k_i = 1$ for all $i \in N$, $l_i = 1$ for all $i \in N$, $a_{ij} = 1$ for all $(i, j) \in A^R$, $l = 1$ and $r = 6$, allowing us to use the constraint constructed in Section 3.4.1.

Climbing the ladder constraints are included to enforce that interdictions start at nodes lower in the network before nodes higher in the network can be interdicted, similar to how law enforcement would investigate a drug trafficking network. The number of interdicted neighbors for the climbing the ladder constraints is determined as per Malaviya et al. [75]. The cost for interdicting a node in the climbing the ladder constraint is determined by aggregating the targeting requirement and arrest requirement values from Malaviya et al. [75]. We again note that interdicting is not solely law enforcement arresting participants, but can be any sort of intervention by an interested stakeholder. We can more appropriately think of the targetting and arrest requirements as the cost of observing participants and implementing interdictions, respectively.

We limit the solve time of each experiment to two hours, with $\epsilon = 10^{-4}$. If an instance were to exceed the two hour solve time, we use the last iteration completed within the time limit to determine the lower and upper bounds. Additionally, after completing an iteration, we check if the total solve time plus the solve time of the most recent iteration is under two hours, and terminate if it is not. Since the problem grows in size each iteration, we expect that each subsequent iteration will be at least as long as the previous one, thus terminating early if we expect the next iteration to terminate after the time limit would be exceeded. We model the problem in AMPL with Gurobi 9.0.2 as the solver. Experiments were conducted on a laptop with an Intel® Core™ i5-8250 CPU @ 1.6 GHz - 1.8 GHz and 16 GB RAM

running Windows 10.

We first compare solve time with the method described in [129]. Before doing so, we note that the KKT-conditions-based tightening cannot be applied to our problem. This tightening tries to improve the lower bound on the objective value of the upper level problem by having the upper level decision maker (attacker) try to predict the lower level decision maker's (defender's) integer decisions (restructuring decisions), and identifies the optimal continuous variables in the lower level problem. In our problem, for any interdiction plan, a restructuring plan of $z = 0$ is always feasible. Since we are in a minimization problem, the null plan will always be selected for these constraints, resulting in the lower bound given by the KKT-conditions-based tightening being the interdicted flow. If we were to enforce that restructuring needs to occur, we may encounter the scenario where an interdiction plan does not allow for restructuring to occur; this would make the master problem infeasible.

To implement their projection scheme, we extend our w variables to determine the full feasibility of a restructuring plan. For each iteration k, let

$$f^k = \begin{cases} 1 \text{ if the } k^{th} \text{ restructuring plan is feasible for current interdiction decision } y \\ 0 \text{ otherwise.} \end{cases} \tag{3.36}$$

We can enforce that f^k takes the correct value with the following constraint:

$$\begin{aligned} f^k \geq & \sum_{(i,j) \in A^R : z_{ij}^{out,k} = 1} w_{ij}^{out,k} - |\{(i,j) \in A^R : z_{ij}^{out,k} = 1\}| \\ & + \sum_{(i,j) \in A^R : z_{ij}^{in,k} = 1} w_{ij}^{in,k} - |\{(i,j) \in A^R : z_{ij}^{in,k} = 1\}| + 1. \end{aligned} \tag{3.37}$$

With this constraint, if all arcs that were restructured in the k^{th} restructuring plan are able to be restructured, then the sums well be equivalent to the set cardinalities, and thus $f^k \geq 1$. Otherwise, the constraint reduces to $f^k \geq c$ with $c \leq 0$, resulting in $f^k = 0$ being selected due to solving a minimization problem. Additionally, instead of w_{ij}^{in} or w_{ij}^{out} being included in the constraints to indicate the side of the cut each node is on, f^k is included in its place.

Over the 250 instances, our method only fails to solve three instances to optimality within two hours. All three of these instances are instances with the recruitment of new participants and have medium attacker budget levels. For simplicity of comparison, we compare our method against the algorithm of [129] for just the base instances and the

instances with recruitment of new participants.

For the base instances, we observe that, when the algorithm of [129] is able to solve the problem, it typically requires more plans than Algorithm 6 to find the optimal solution. However, when both methods require a similar number of plans needed to solve the problem, our method is generally slower than that of the algorithm of [129]. We suspect this is due to our method needing to explore branch-and-bound nodes where parts of previously visited restructuring plans may be relevant. However, when our method needs less plans to find the optimal solution, we are able to solve the problem significantly faster (e.g., for the majority of instances for Data3, we require well under 20 minutes while their method hits the 2 hour upper bound). Table 3.1 reports the run times and number of plans visited by each method.

For the instances when recruitment of outside participants is allowed, we are able to more clearly see the advantage in our method over that of the algorithm of [129]. While our method is able to solve all but three instances, their method is only able to solve 9 of the 50 instances within the specified time limit. We further observe that an extremely large percentage (38 out of 41) of our run times are under an hour and a large percentage (33 out of 41) are under half an hour for instances where they fail to solve the problem to optimality. Additionally, their method visits a significantly larger number of plans while still leaving a large relative optimality gap. Table 3.2 compares the run times and number of plans visited by both methods. These instances are marked with an asterisk in Table 3.2. We additionally report the relative optimality gap of these instances as the difference between the lower and upper bound divided by the upper bound in Table 3.3. These results demonstrate the significant importance of our cuts that use partial information about previous interdiction plans and the restructuring response.

3.6 Computational Results: Application-based Analysis

In our tests, we examine the limitations of applying the interdiction plan determined to be optimal by solving the traditional MFNIP. This will help determine the importance of accounting for restructuring in the context of disrupting drug trafficking networks. In each network, we observed that the optimal solution value of MFNIP-R is less than that of the optimal restructuring to MFNIP's optimal interdictions. In most instances, the increase from the MFNIP solution to the MFNIP-R solution is less than half of the increase from the MFNIP solution to the optimal restructuring after the MFNIP solution. We demonstrate the

Table 3.1: Comparison of run times (seconds) and number of plans visited of algorithm 6 and the algorithm of [129] in base restructuring instances

Budget	Data1, Alg1	Data1, [129]	Data2, Alg1	Data2, [129]	Data3, Alg1	Data3, [129]	Data4, Alg1	Data4, [129]	Data5, Alg1	Data5, [129]
50	**110.875 (3)**	561.094 (6)	**10.938 (3)**	73.079 (7)	**25.236 (3)**	7200 (18)*	**18.86 (3)**	103.125 (6)	**139.688 (3)**	1282.75 (8)
60	**18.844 (3)**	55.328 (5)	**5.672 (2)**	38.75 (4)	**521.078 (8)**	7200 (17)*	**23.75 (3)**	399.781 (10)	**70.422 (2)**	6132.687 (14)
70	**12.781 (2)**	115.922 (5)	**22.968 (3)**	181.234 (7)	**449.516 (7)**	7200 (17)*	**51.765 (4)**	260.844 (8)	**177.031 (3)**	2139.485 (8)
80	48.313 (3)	**43.937 (3)**	47.078 (3)	**41.594 (3)**	**1316.953 (7)**	7200 (15)*	**104.438 (4)**	2388.859 (12)	**170.422 (3)**	7200 (14)*
90	**45.203 (2)**	118.797 (3)	**122.359 (3)**	155.86 (5)	**688.641 (5)**	7200 (13)*	**139.515 (4)**	533.484 (7)	**56.063 (2)**	2843.36 (11)
100	61.188 (2)	**33.953 (2)**	95.031 (3)	**75.468 (4)**	**69.656 (2)**	7200 (12)*	722.641 (5)	**282.453 (5)**	**210.328 (3)**	7200 (9)*
110	**80.078 (2)**	179.234 (3)	**3.219 (2)**	4.438 (3)	**118.578 (2)**	7200 (11)*	**409.906 (4)**	569.954 (5)	**312.594 (3)**	7200 (11)*
120	378.562 (3)	**240.813 (3)**	0.156 (1)	**0.141 (1)**	**90.485 (2)**	7200 (12)*	968.485 (6)	**910.281 (7)**	**97.468 (2)**	4206.719 (10)
130	636.907 (3)	**329.656 (3)**	**0.219 (1)**	0.578 (2)	**1893.594 (7)**	7200 (13)*	**537.984 (6)**	884.829 (7)	**219.719 (3)**	7200 (12)*
140	14.343 (2)	**3.813 (2)**	0.141 (1)	**0.094 (1)**	**4832.516 (12)**	7200 (13)*	62.437 (4)	**9.546 (3)**	**758.61 (5)**	3095.734 (9)

Table 3.2: Comparison of run times (seconds) and number of plans visited of algorithm 6 and the algorithm of [129] in instances with recruitment

Budget	Data1, Alg1	Data1, [129]	Data2, Alg1	Data2, [129]	Data3, Alg1	Data3, [129]	Data4, Alg1	Data4, [129]	Data5, Alg1	Data5, [129]
50	**55.5 (3)**	7200 (18)*	**37.75 (4)**	7200 (35)*	**211.891 (6)**	7200 (12)*	**70.407 (4)**	7200 (29)*	**161.484 (5)**	7200 (17)*
60	**70.172 (4)**	7200 (16)*	**48.11 (4)**	7200 (25)*	**62.687 (4)**	7200 (15)*	**37.453 (2)**	7200 (20)*	**166.766 (4)**	7200 (14)*
70	**344.578 (6)**	7200 (17)*	**45.953 (3)**	7200 (13)*	**114.157 (4)**	7200 (14)*	**129.265 (4)**	7200 (24)*	**520.172 (5)**	7200 (13)*
80	**575.875 (8)**	7200 (18)*	**163.906 (4)**	7200 (14)*	**1421.5 (6)**	7200 (12)*	**1463.5 (7)**	7200 (15)*	**2379.765 (8)**	7200 (13)*
90	**1799.312 (5)**	7200 (12)*	**179.922 (5)**	7200 (9)*	**3364.844 (6)**	7200 (10)*	7200 (8)*	7200 (12)*	**1892. 297 (7)**	7200 (13)*
100	**1543.235 (5)**	7200 (11)*	**962.062 (5)**	7200 (5)*	**720.406 (4)**	7200 (10)*	**2831.515 (6)**	7200 (11)*	**2406.187 (7)**	7200 (13)*
110	**1288.687 (5)**	7200 (9)*	**260.438 (4)**	7200 (10)*	**705.374 (4)**	7200 (10)*	**3586.438 (7)**	7200 (11)*	7200 (8)*	7200 (13)*
120	**2373.344 (6)**	7200 (10)*	**527.531 (5)**	2069.875 (10)	**851 (4)**	7200 (9)*	**718.875 (3)**	5169 (10)	7200 (8)*	7200 (13)*
130	**161.641 (2)**	1394 .063(6)	**67.265 (4)**	152.109 (11)	**595.157 (4)**	7200 (5)*	**4260 (8)**	5744.562 (11)	**644.422 (4)**	7200 (11)*
140	28.375 (2)	**16.547 (2)**	0.141 (1)	**0.11 (1)**	**249.781 (3)**	3094.141 (9)	**714.016 (5)**	921.797 (6)	**457.531 (3)**	7200 (11)*

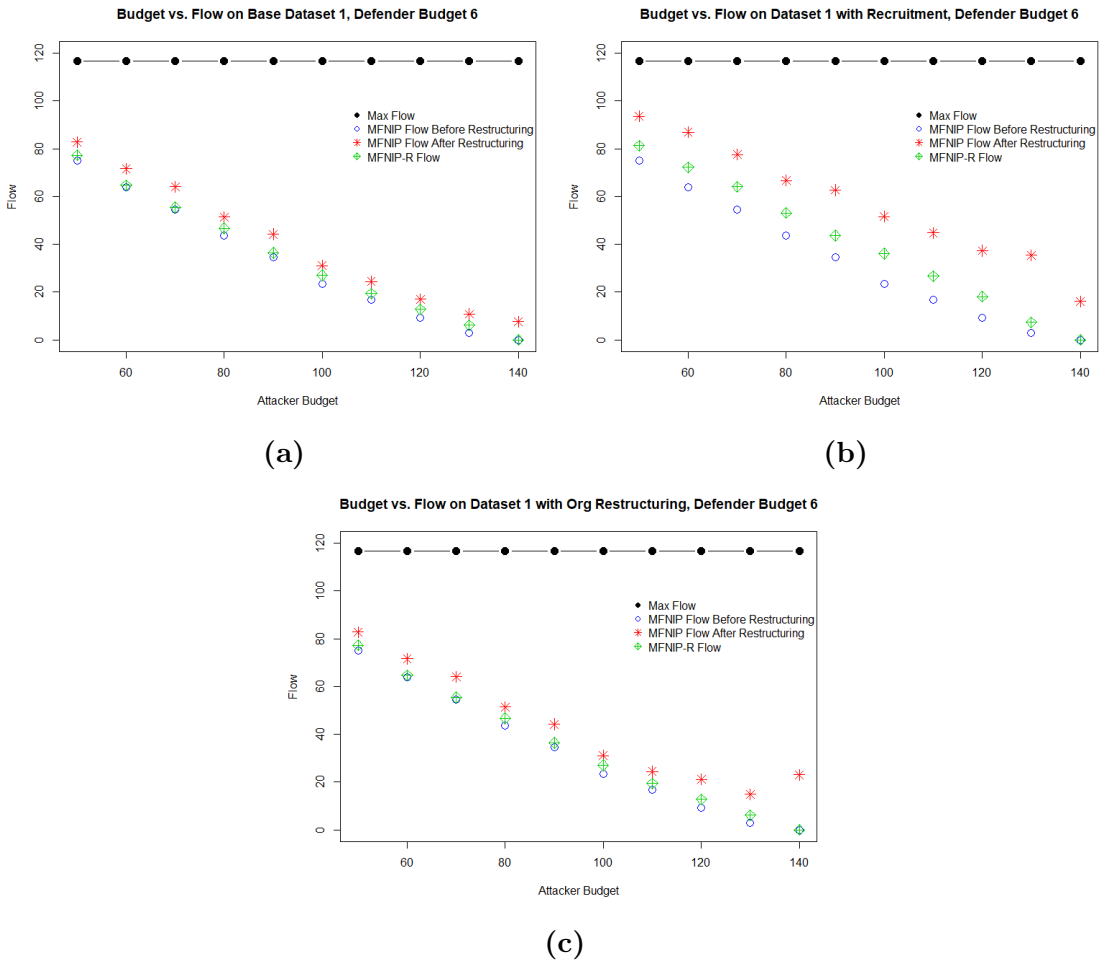
Table 3.3: Relative optimality gaps of unsolved instances of algorithm 6 and the algorithm of [129] in instances with recruitment

Budget	Data1, Alg1	Data1, [129]	Data2, Alg1	Data2, [129]	Data3, Alg1	Data3, [129]	Data4, Alg1	Data4, [129]	Data5, Alg1	Data5, [129]
50	-	12.764%	-	27.447%	-	15.238%	-	12.993%	-	8.915%
60	-	18.710%	-	33.073%	-	14.314%	-	23.405%	-	15.194%
70	-	21.093%	-	33.642%	-	18.612%	-	17.951%	-	20.278%
80	-	26.672%	-	39.175%	-	19.186%	-	38.355%	-	23.868%
90	-	32.142%	-	56.361%	-	20.347%	0.637%	38.368%	-	31.891%
100	-	36.823%	-	69.444%	-	28.804%	-	35.693%	-	37.486%
110	-	29.415%	-	44.209%	-	39.164%	-	10.904%	0.929%	46.313%
120	-	39.628%	-	-	-	46.850%	-	-	7.865%	52.938%
130	-	-	-	-	-	63.277%	-	-	-	59.492%
140	-	-	-	-	-	-	-	-	-	65.755%

differences in these values with figures of one of the data sets, as the patterns we can observe from it are representative of all five data sets. Figure 3.4a compares flow values against the attacker budget in our base instance, which only allows for arcs to be restructured from a node in a given tier to a node in a tier immediately above or below it. Figure 3.4b presents this comparison on instances with recruitment of new participants, i.e., new nodes can be added to the network, and Figure 3.4c presents this comparison on instances with organization restructuring, i.e., new nodes can be added and arcs can skip tiers. In each plot, we include the original maximum flow through the network (Max Flow), the maximum flow from the MFNIP solution (MFNIP Flow Before Restructuring), the maximum flow with the optimal restructuring after MFNIP interdictions (MFNIP Flow After Restructuring), and the maximum flow determined by MFNIP-R (MFNIP-R Flow).

We can see that there is a significantly larger increase in flow after restructuring when recruitment of new participants is allowed. This supports the idea in criminology literature that interdicting at lower levels does not decrease the flow as much as is commonly assumed, since lower level participants are replaceable. Additionally, we note that the flow of MFNIP after restructuring with organization restructuring at an attacker budget of 140 is larger than that flow with a budget of 130. Because MFNIP does not account for restructuring, the interdiction plan identified using a budget of 140 was able to initially disrupt more of the flow, but also allowed a different set of restructurings to occur. These restructurings resulted in significantly more flow in the resulting network. Even though we have a better capability to disrupt the network, this demonstrates that how we disrupt the network is just as important when we consider its ability to restructure after the interdiction. We also observed that the defender budget constraint was rarely fully used in the optimal bilevel solutions, indicating that we were also essentially solving the problem where we allow unlimited restructuring but constrain it based on where we have interdicted.

We note that there is very little difference in the flow values between the base data set and the adaptation with organizational restructuring. This is due to the fact that the interdiction decisions do not allow for these additional arcs to be allowed to be restructured by not disrupting high enough in the network. Ideally, we would want interdiction decisions to disrupt higher in the network, as we can see that disrupting lower in the network appears to reduce the flow significantly, but the presence of recruitable participants outside the network can drastically increase the flow after restructuring. To measure the impact of interdicting



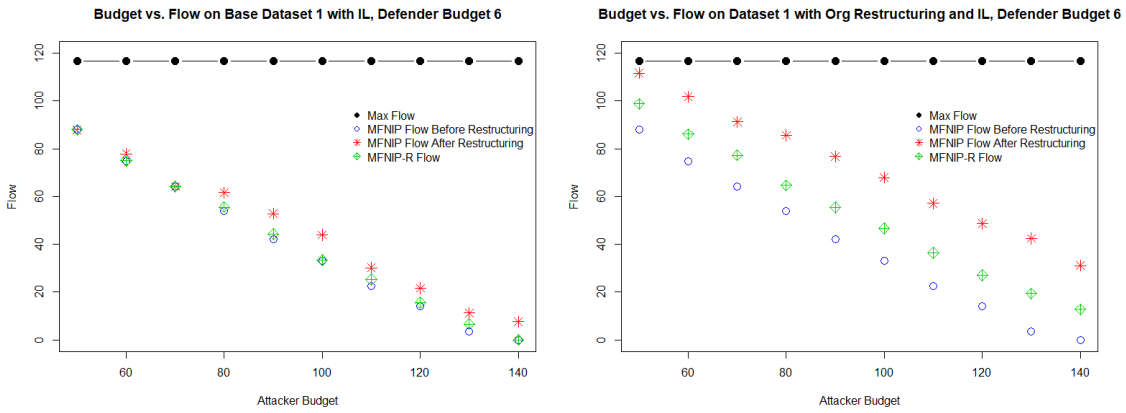
(a) (b)

(c)

Figure 3.4: Bilevel problem objective values over varying attacker budgets where (a) is the base instance, (b) includes recruitment, and (c) includes organizational restructuring

nodes that are in higher tiers, we include the constraint on interdicting leadership (see Appendix A for more details). Figure 3.5a plots the base instance flows with the interdicting leadership constraint, and Figure 3.5b plots the flows with organizational restructuring and the interdicting leadership constraint.

From here we can clearly see the impact of organizational restructuring. In the base instance, there was a relatively small gap between the MFNIP flow and the MFNIP flow after restructuring. However, when we allow for organization restructuring, there is a significantly larger gap, and the MFNIP-R solution is larger than in the base instances. However, we are still able to interdict in a way that mitigates the large gains in flow from restructuring after the interdictions proposed by MFNIP. This again demonstrates that how we interdict the



(a) **(b)**

Figure 3.5: Bilevel problem (with interdicting leadership) objective values over varying attacker budgets where (a) is the base instance and (b) includes organizational restructuring

network is much more important when we have to consider how the network will restructure after our interdictions.

In most MFNIP optimal solutions, the highest tier with interdictions is the safe house tier, the third tier from the bottom. This is due to the differences in capacities between the tiers; the flow can be decreased more drastically by interdicting nodes in the lower tiers than by using the equivalent amount of resources to interdict a node higher in the network. This is further exacerbated by the climbing the ladder constraints; it is much more costly to interdict higher in the network, further increasing the value of interdicting lower in the network. However, when recruiting outside participants is considered, we can see that more flow can be recovered than in the base instances. By interdicting more participants higher in the network, we expect that we will be able to better disrupt the network functionality, since these participants are more difficult to replace than lower level participants due to there being fewer participants in this tier. These gains from recruitment can be mitigated if efforts were made to limit lower level recruitment. By dedicating resources in ways that would decrease vulnerabilities of people to be recruited into the network, we can limit the ability of the network to replace these lower level participants, and focus on the disruption of higher level participants. Since the optimal flows in our base model are lower than the optimal flows in the model with recruitment, investing in communities to prevent recruitment into these networks would be more beneficial than reacting to recruitment of new users and dealers.

We must be careful when we interdict higher in the network when organizational restructuring is considered, as this type of restructuring allows for a significant amount of initial flow to be regained. By dedicating resources to determining which participants are most essential and least replaceable, we can exploit the vulnerabilities of the network structure to more effectively disruption the operations of the network.

3.7 Conclusion

We define the max flow network interdiction problem with restructuring, where the defender is able to introduce new arcs to the network after interdiction decisions have been made. We demonstrate that, by not considering restructuring, interdictions can result in an *increase* in flow, not a decrease as expected. Constraints on the decisions of the defender were defined based on modeling city-level drug trafficking networks. We also introduce the concepts of restructuring to recruit participants outside of the network and organizational restructuring, where participants are promoted and participants from other organizations can be recruited to a different organization. To solve this problem, we also improve upon the column-and-constraint generation procedure by utilizing partial information from infeasible restructuring plans.

For generated data on city-level drug trafficking networks consisting of 6 tiers, we observe that the interdictions often disrupt up through the third lowest tier of the network, similar to the traditional MFNIP solution, but attain flow values less than that of the restructuring response to the optimal solution of MFNIP. This illustrates the importance of including restructuring in the network interdiction decision and makes interdiction models more aligned with reality. Without specifically requiring it, we are still mostly interdicting the "replaceable" participants in the drug trafficking network. We suspect that, to inflict lasting damage on the operations of the network, we need to target higher up in the network. While we are able to assert more control over the flow; the optimal solution is still to not interdict the tiers of the network that are harder to replace. This is due to the fact that, for equivalent resources, flow decreases more when we interdict more nodes in lower tiers than fewer nodes in higher tiers, because of the additional costs from the climbing the ladder constraints.

We also observe that recruiting new participants and organizational restructuring in drug trafficking networks each result in a significant increase in flow after restructuring.

While organizational restructuring may not occur to the extent we model, recruitment of new participants is a known issue. Our analysis suggests it is best to not only focus on disrupting the current network, but also prevent the network from recruiting participants.

CHAPTER 4
GENERATING SYNTHETIC BUT REALISTIC SEX TRAFFICKING NETWORKS FOR MODELING DISRUPTIONS THROUGH TRANSDISCIPLINARY AND COMMUNITY BASED ACTION RESEARCH

4.1 Introduction

In this chapter, we propose a novel network generator that outputs network configurations that are representative of domestic sex trafficking networks in the real world and that account for realistic variation among operations. This generator takes the number of desired trafficking operations for use in analysis as an input, then produces the operational and social connections between the participants in the network. We also describe the process by which our generator was developed. Our generator is the product of triangulating multiple data sources, including case file analysis, interviews with people who have domain expertise, and validation with a survivor-centered advisory group. The networks produced by the generator can be used for various analytical models without the need for researchers to collect and validate their own data. We note that the development process for the network generator is iterative. The more the generator is used alongside domain experts, the more it can be refined, expanded and further validated for improved accuracy.

Additionally, we propose a novel conceptualization of flow for sex trafficking networks; we model flow as the ability of the traffickers to control their victims. Thus, the maximum flow through the network is the total number of victims the traffickers victimize at a particular moment in time. We demonstrate how max flow network interdiction can be applied to these networks and discuss policy recommendations for when the traffickers are able to restructure their operations after the interdiction decisions have been implemented.

This chapter is organized as follows: Section 4.2 outlines our objectives when developing the network generator and the process of triangulating data sources to develop the generator. Section 4.3 addresses the assumptions of the network generator and analyzes sample outputs.

This chapter has previously appeared as: D. Kosmas, C. Melander, E. Singerhouse, T.C. Sharkey, K.L. Maass, K. Barrick, and L. Martin, *Generating Synthetic but Realistic Human Trafficking Networks for Modeling Disruptions through Transdisciplinary and Community-Based Action Research*, preprint, arXiv:2203.01893 [cs.SI], 2022. Submitted to IISE Trans.

Section 4.4 focuses on a responsible use case study of the generator that focuses on applying a novel network interdiction model to disrupt the operations of generated sex trafficking networks. Section 4.5 discusses limitations of our network generator and proposes directions for future qualitative research to improve the applicability of the network generator. We discuss our conclusions in Section 4.6.

4.2 Development of the Network Generator

4.2.1 Goals

The generator focuses on the social and operational connections of the people within domestic sex trafficking networks with specific operational characteristics. In this chapter, we recognize that these networks are abstractions of the true operations of sex trafficking. Therefore, the proposed network generator cannot appropriately capture all of the complexities of the lived experiences of trafficking victims and survivors, nor can it account for the human rights abuses and violence that occur in sex trafficking networks. This is a limitation of any analytical approach to understanding the issue of sex trafficking.

For our network generator, we focused on the social and operational connections of the participants in domestic sex trafficking networks with certain operational characteristics. The nodes represent the people in the network, and arcs represent the connections between them. The participants we consider are the traffickers, so-called "bottoms", and victims. The term "bottom" is used in the literature and in some operations to refer to someone who was a victim, but has gained trust and responsibility from the trafficker and thus has additional responsibilities within the trafficking operation [126]. They might be viewed as the highest ranking former victim and function as a sort of right-hand person in the operation. However, there is a gray area in that bottoms may experience varying degrees of force and coercion since there is always a chance they will lose this status and be forced to trade sex. Likewise, a trafficker can expect or force some bottoms to exploit other victims [105, 10, 126]. For purposes of the generator, victims are split into two groups based on age, either minor or adult. We do this because published research and insights from our survivor-centered advisory group suggest that there are operational differences in how traffickers interact with minors versus adults due to developmental differences and a legal distinction where trafficking of minors results in much higher penalties [78].

In order to clarify the distinction between the activities of a specific trafficker and the

activities of all traffickers, we additionally define the distinction between an operation and network. Henceforth, when we refer to an operation, we are referring to a single trafficker, their bottom (if they have one), and their victims. The network refers to all operations generated, where the number of operations is a user-specified input. We can view the network as all operations in a given geographical region. We also want to provide a distinction to connections that are necessary for the function of a trafficking operations, as opposed to connections that are purely social, although we recognize that social connections may help further the activities of a trafficking operation. We define an operational arc as an arc that is necessary for the functions of the trafficking operation. An example of this is an arc between a trafficker and a victim, as that connection represents that the trafficker is able to control the victim. We define a social arc as an arc that does not necessarily have a function associated with them. An example of this is an arc between two victims. Such a connection may not be necessary for the act of sex trafficking.

4.2.2 Transdisciplinary Research Approach: Creation and Validation

We sought to utilize different data sources to guide the development of the network generator. All data sources focused on trafficking operations in the Midwest region of the USA. For this first phase of development of the network generator, data analysis focused on smaller operations with only one trafficker. Table 4.1 denotes the number of each type of data source.

Table 4.1: Data sources synthesized to help produce the network generator

Data Source	Number
Publicly Available Federal Case Files	13
Key Informant Interview	10
Secondary Analysis of Interviews from Previous Studies [80, 81]	246

We began the development of our network generator by first investigating publicly available federal case files of domestic sex trafficking cases. These case files consisted of relevant information needed to prosecute the trafficker(s), include the people involved, locations visited, online presence, and connections between them. However, these case files only contain the information necessary to prove a crime has been committed, which may omit key details regarding the network structure [33].

The data contained in these case files focused on the victims identified by law enforcement and the actions performed by the trafficker that would be necessary to prove that sex trafficking had been committed, and anything needed to perform those actions. Since the case files focused on proving a crime had been committed, they often had too much and too little information. For example, the case files included details connecting people to cell phones. While communications between the people using the phones is important for modeling connections between individuals, capturing how many phones each person uses and how phones move from person to person would be difficult to generalize in the network generator. Additionally, only considering the details necessary to prove the elements of the crime provides an inaccurate picture of the full number and roles of people involved in the network.

Figure 4.1 provides a visualization of all connections between people, places and things in one of the case files. The participants in the network are split into three tiers. The trafficker tier consists of the traffickers, the criminal associate tier consists the trafficker's bottom and any associates of the trafficker who assisted in criminal activity, and the person trading sex tier consists of the people who were coerced to participate in sexual activity. The "item" tier includes phones, vehicles, and online advertisements. The location tier includes any location visited by participants in the network, such as where the trafficker met the people they trafficked. The intervener/sex buyer tier includes anyone who interacted with the network but was not a direct participant, such as law enforcement and sex buyers. This figure demonstrates the incompleteness of data from case file analysis. Most notably, there was no record in the case file connecting the trafficker (T1) with their bottom (B1), which is a connection we would expect to see and was commented on by our advisory group when this network was presented to them. However, such a connection would not be necessary to prove that the person trading sex (PTS1) had been victimized by either T1 or B1. Moreover, these case files spanned a short duration of time, thus only capturing a small snapshot of the trafficker's operation in that moment in time. In the case of the network from Figure 4.1, the recorded operations only spanned a week.

The second source of data we investigated was transcripts of interviews conducted with people who have first-hand knowledge of sex trafficking networks, including survivors, support people, and law enforcement. These interviews guided how to determine the size of networks, as well as how to determine social connections between victims.

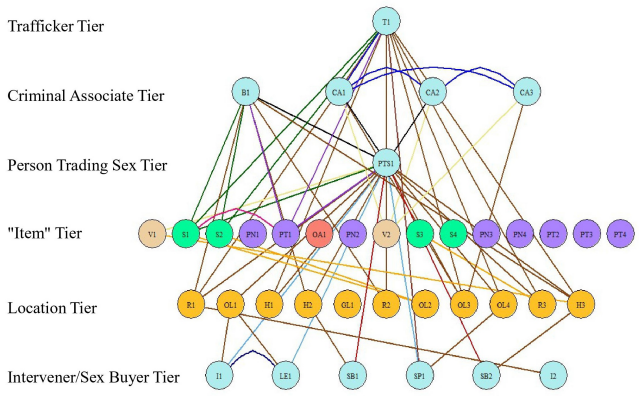


Figure 4.1: Sample network from case file analysis

From triangulating these sources, we devised a basis for network generation. We devised rules from consistencies across multiple data sources, such as the distribution of number of victims within an operation, and used statistics and qualitative descriptions to estimate parameters for these rules. Once all necessary parameters were determined, we generated a set of networks to review with domain experts on the research team, as well as a list of questions and observations to discuss. We reviewed these networks and engaged in discussions to clarify what seemed to be accurate based on their expertise. After the discussions, the domain experts were provided the outputs to further discuss. They wrote a memo that critiqued the outputs with the known literature on domestic sex trafficking and then discussed how to improve the generator. Some topics the domain experts commented on included adjusting parameters for determining whether a bottom is in the network and how victims are connected to traffickers and bottoms. Adjustments to the rules and parameters were made based on the suggestions in the memo. This process of reviewing networks produced by the generator with the domain experts was repeated multiple times. Figure 4.2 displays the probability distribution of the number of victims in a trafficking operation after it was critiqued by the domain experts and adjusted to better reflect their knowledge.

After converging to output networks of the generator that reflected the knowledge of the domain experts on the research team, we then presented the networks to a survivor-centered advisory group. As part of the research team's approach to transdisciplinary research, the advisory group was familiarized with the concept of networks prior to these presentations. The basis for the network generator was presented to validate if the underlying rules for network generation seemed accurate based on their knowledge, and sample output networks

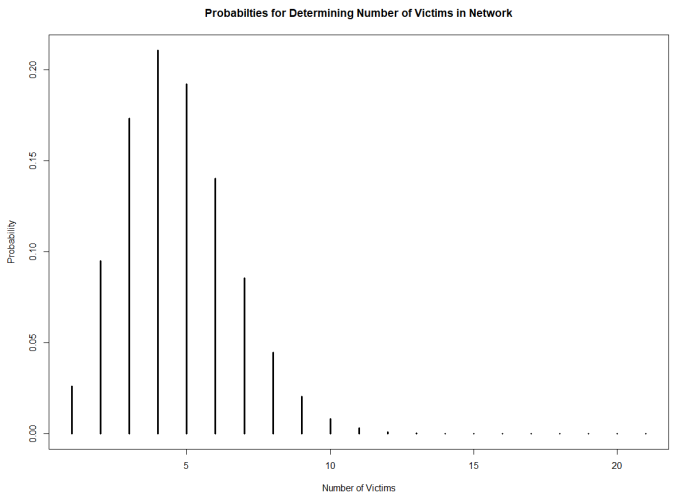


Figure 4.2: Probability distribution of number of victims in a trafficking operation, validated with domain experts

were presented for members of the advisory group to discuss. The researchers also solicited feedback on if there were other factors that should be incorporated into the network generator, both for immediate adjustments and directions for future versions of the generator. After this presentation, adjustments were made, and new output networks were presented to domain experts on the research team. In particular, discussions with the advisory group led to a new method to determine social connections between victims. To the best of our knowledge, very little literature explores social connections between victims [32]. Conversations with the advisory group shed some light on these connections, such as smaller groups of victims being housed together and separated from the rest of the victims. After implementing this new method, the domain experts were able to refine the choice of parameters for this method.

4.2.3 Details of the Network Generator

Our network generator produced a network of single trafficker operations, where the number of traffickers is an input n_T. For each operation i, we first generate the number of victims n_{vi} a trafficker has in their operation. The distribution we sample from is based on statistics gathered and validated by the domain experts. Based on the number of victims, we randomly determine if there is a bottom in the network. The probability of a bottom in the network grows significantly with respect to the number of victims, with a bottom almost surely present when there are at least six victims. If it is determined there is a bottom in

the network, a new node is added to be the bottom, as the statistics collected for number of victims did not include a bottom as a victim. These observations were made based on a secondary analysis of previously collected interview data [81, 80].

We next determine how a trafficker is managing the victims within their operation. We do so by partitioning the victims into "pods" [85]. Pods can be thought of as groups of victims who were recruited roughly at the same time, or live in the same location. Each victim in a given pod is connected to every other victim in the same pod, i.e., a pod forms a clique in the operational network. We generate all feasible partitions of the number of victims where each part has at most six victims, then randomly select from this set [85]. Feedback from the advisory group provided insights on how to determine the probability w of selecting different partitions, as well as the size of the pods. Let $\mathcal{P}$ be the set of all feasible pods. We say $P \in \mathcal{P}$ is a pod structure, where $P = (p_1, \ldots, p_{|P|})$, and p_i is the number of victims in the i^{th} pod. When there is more than one pod, we randomly determine the age group (i.e., minor or adult) of the victims for the entire pod, as opposed to determining the age group for victims individually. The exception to this is when a pod consists of two victims, which we allow for one victim to be a minor and one victim to be an adult. This is representative of a situation where a pair of victims may be family members (e.g. mother and daughter) or where there is a parental-type relationship. These choices regarding ages and pods were suggested by the advisory group. Algorithm 7 formalizes how pods are generated and how victims are classified by age.

We then determine how the trafficker (and bottom) interact with the pods. If there is a bottom, we first determine how the victims are connected to the trafficker and bottom, as all victims are not necessarily connected to both the trafficker and the bottom. We randomly determine which pods are connected to the trafficker, then which pods are connected to the bottom. Similarly, if the trafficker (or bottom) is not randomly assigned any pods, they will be assigned to the largest pod. The probability of a pod being assigned to the trafficker and bottom varies based on the age of the victims in the pod. This is due to a trafficker potentially not wanting to have direct contact with a victim who is a minor, as prosecuting a sex trafficking case is easier if the victim is a minor, and the minimum punishments are more severe [78]. If there is no bottom, all victims are connected to the trafficker. If there is a single pod, then age and connections to trafficker and bottom are determined individually, as if each victim were in their own pod.

Algorithm 7 Generation Procedure for Pods in a Single Trafficker Operation ($GeneratePods(i, n_{vi})$)

Initialize: operation number i, number of victims n_{vi}.
if $n_{vi} > 1$ **then**
  Step 1a. Generate set of feasible partitions $\mathcal{P}$ of n_{vi}.
  Step 1b. Randomly select from $P \in \mathcal{P}$ using weighted probabilities w.
else
  Step 1c. Set $P = \{1\}$.
end if
if $|P| \geq 2$ **then**
  for $j = 1 : |P|$ **do**
    Step 2a. Randomly generate age group of pod of victims $p_j \in P$.
  end for
else
  for $j = 1 : n_{vi}$ **do**
    Step 2b. Randomly generate age group of victim $v_j \in P$.
  end for
end if

We then expand upon the social network amongst victims within the operation. Partitioning the victims into pods provides an initial set of arcs for the social network amongst victims. We then determine any social connections between victims in different pods. For each pair of victims in different pods, we randomly determine if an arc between them should be added to the network. The probability of an arc being added between them is dependent on the age of the victims, since minor victims tend to be recruited into trafficking via their social network [78]. Algorithm 8 formalizes the procedure of generating a single trafficker operation.

This procedure is repeated for the number of operations desired. After all operations are generated, we generate the social network amongst traffickers. From Dank et al. [39], we know that traffickers are connected to share information about profitable locations and law enforcement activity. Since little further is known about how traffickers are connected socially, we use the Watts-Strogatz model to generate the social network amongst traffickers [121]. The Watts-Strogatz model is a random graph generation model that is often used for social networks since it produces networks with the "small world phenomena," which indicates that the shortest paths between pairs of nodes include a small number of arcs and that nodes tend to be grouped into clusters, with a larger number of arcs between nodes in the same cluster than between nodes in different clusters. After the trafficker social network

Algorithm 8 Generation Procedure for a Single Trafficker Operation ($GenerateOp(i)$)

Initialize: operation number i.
Step 1. Generate number of victims n_{vi} for operation i.
Step 2. Randomly determine if operation i has a bottom b.
Step 3. Partition victims into pods and determine age groups with $GeneratePods((i, n_{vi})$.

if Operation i has a bottom **then**
  if $|P| \geq 2$ **then**
    for $j = 1 : |P|$ **do**
      Step 4a. Randomly determine if pod of victims p_j is directly connected to the trafficker, bottom, or both.
    end for
  else
    for $j = 1 : n_{vi}$ **do**
      Step 4b. Randomly determine if victim v_j is directly connected to the trafficker, bottom, or both.
    end for
  end if
else
  for $j = 1 : n_{vi}$ **do**
    Step 4c. Connect victim v_j to the trafficker.
  end for
end if
if $|P| \geq 2$ **then**
  for $j = 1 : (|P| - 1)$ **do**
    for $k = (j + 1) : |P|$ **do**
      for $u \in p_j$ **do**
        for $v \in p_k$ **do**
          Step 5. Randomly determine whether or not victims u and v are connected.
        end for
      end for
    end for
  end for
end if

is generated, we then generate social connections between victims in different trafficking operations. Again, we impose a higher likelihood of two victims who are minors being connected over adult victims. Parameters regarding social connections were validated by the domain experts and advisory group. We now formalize our network generator in Algorithm 9.

Algorithm 9 Network Generator for Networks of Single Trafficker Operations

Initialize: number of traffickers n_T.
for $i = 1 : n_T$ **do**
  Step 1. Generate single trafficker operation i with $GenerateOp(i)$.
end for
Step 2. Generate trafficker social network.

4.3 Outputs

We now present sample output networks from our domestic sex trafficking network generator and compare them to sex trafficking networks constructed in previous research. Figure 4.3 displays five sample operations generated by the network generator. Trafficker nodes are squares, bottom nodes are triangles, and victims nodes are circles. Nodes representing victims who are adults are a darker shade of gray than nodes representing victim who are minors. Arcs are solid if they are representing an operational arc (e.g. trafficker to victim), or dashed if they are representing a social arc. We include arcs between victims in the same pod as an operational arc.

In these five operations, all but one operation has a bottom. In 25 generated outputs, 20 operations have bottoms, and every operations with more than 5 victims had a bottom. Of the eight operations with 4 victims, only one operation does not have a bottom. Of the four operations with 3, half of them have a bottom. Both networks with 2 victims do not have a bottom. Additionally, of the 25 operations, 5 have one pod, 16 have two pods, 2 have three pods, and 1 has four pods. In the operation with four pods, three of them are isolated victims. The median pod size has 2 victims, with a maximum pod size of 5 victims.

We compare centrality measures of the output operations in Figure 4.3 against those listed in Cockbain [32]. We exclude Operation Retriever, as the number of victims in that network is significantly higher than the other networks in the text. We note that this analysis is comparing operations from two different locations (Midwest USA versus UK), and so trafficking networks may be structured differently between the two locations.

The number of victims in operations produced by the network generator is similar to the number of victims in the operations investigated by Cockbain. Additionally, the arc densities and degree centralization scores of the synthetic operations all fall in the range of arc densities and degree centralization scores of the real operations. Excluding operation 4,

Centrality scores can only be calculated in networks with at least three nodes.

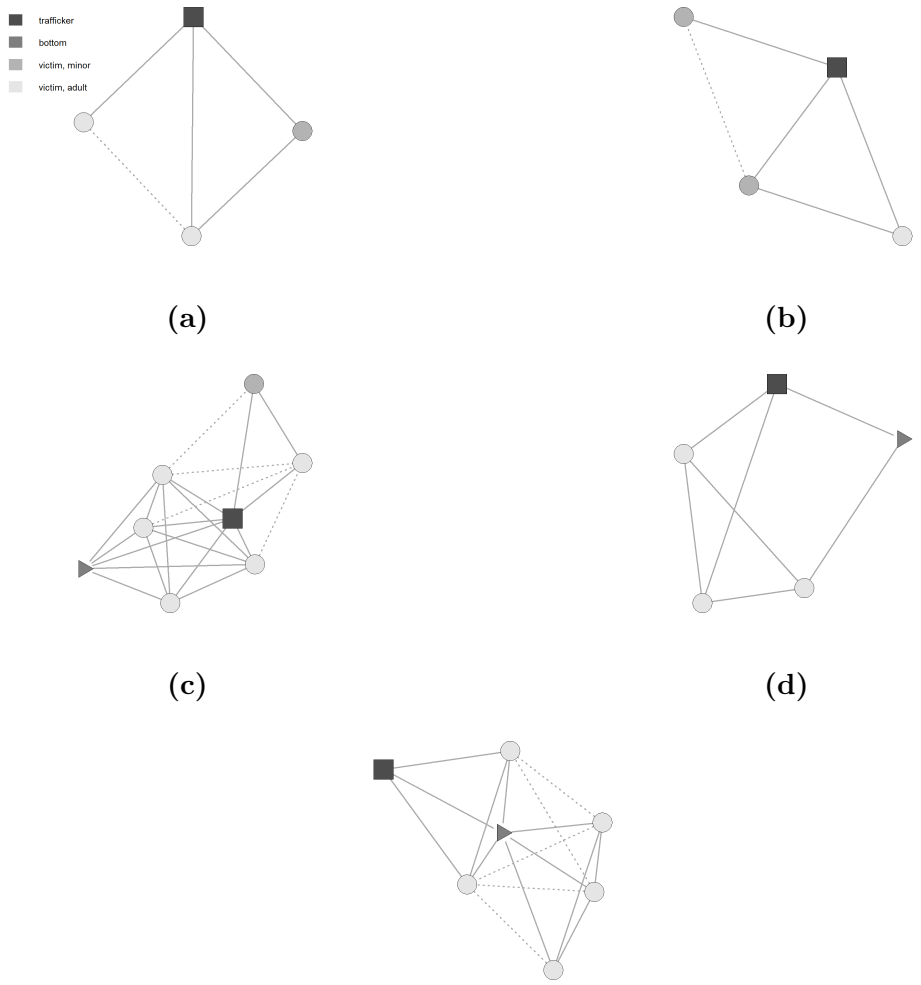


Figure 4.3: Sample operations within a generated sex trafficking network for (a) operation 1, (b) operation 2, (c) operation 3, (d) operation 4 and (e) operation 5

operation Engage, and operation Central, which are all complete networks, the arc densities of the synthetic networks are higher than that of operations Span and Chalice, while the degree centralization scores of the synthetic networks are lower than the real networks. This is likely due to the pod structure in the synthetic networks. In operations with larger pods, the arc density will be larger and the degree centralization will be smaller. The pod structure also likely explains the significant difference in the betweenness centralization scores between the synthetic and real networks. While operation 3 has a betweenness centralization score similar to operations Span and Chalice, operations 1 and 2 have betweenness centralization

Table 4.2: Comparison of centrality measures between operations produced by the network generator and the operations studied in Cockbain [32]

Operation	Number of Victims	Arc Density	Degree Centralization	Betweenness Centralization
1	3	0.6667	0.3333	1
2	3	0.6667	0.3333	1
3	6	0.7333	0.2667	0.2
4	3	1	0	0
5	5	0.9	0.1	0.0278
Engage	2	1	n/a	n/a
Central	4	1	0.0	0.0
Span	5	0.5	0.4	0.2
Chalice	6	0.4	0.6	0.3

scores of 1. This is because these two networks have the same structure: one pod with two victims, and one isolated victim, with a single arc from the isolated victim to one victim in the pod. These results suggest that although there is variation in the synthetic generators, they do share similarities with those produced through case file analysis on sex trafficking networks. This is important as the generator is able to create more synthetic networks for OR analysis.

Once the generator has produced the desired number of operations, the social network amongst traffickers is generated to create the full sex trafficking network. Figure 4.4 displays a sex trafficking network with the five operations in Figure 4.3. At this point, OR models, such as network interdiction could be applied to the full network with appropriate modifications to the data.

4.4 Responsible Use Case Study: Network Interdiction

We now present a case study on how an output of the network generator can be used as data for OR tools. We consider interdiction prescriptions via the max flow network interdiction problem (MFNIP) [127]. In a network $G = (N, A)$, with source node $\alpha \in N$ and sink node $\omega \in N \backslash \{s\}$, and capacities on the arcs $u : A \to \mathbb{R}_{\geq 0}$, the max flow problem seeks to find the total amount of flow from s to t such that the flow on each arc is at most the capacity of that arc, and the amount of flow into a node is the same as the amount of flow leaving that node. The max flow network interdiction problem converts that problem to a two player game, where one player, known as the attacker, seeks to minimize the maximum flow through

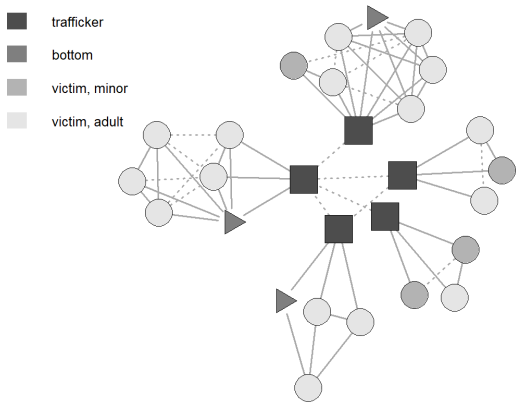


Figure 4.4: Sample sex trafficking network produced by the network generator

the network by choosing a subset of arcs to reduce their capacity to zero, subject to a budget and other constraints. The other player, the defender, then operates the network as per the max flow problem. Max flow network interdiction has successfully been applied to disrupting illicit drug trafficking networks [9, 67, 75, 109], and has been identified as an analytical tool to help address sex trafficking [110]. Some max flow network interdiction models have been proposed for disrupting human trafficking [83, 119]. We apply the model explored in Chapter 3, max flow network interdiction with restructuring (MFNIP-R), to sample outputs of our network generator. Their model accounts for how drug traffickers will respond to disruption efforts. This response is also a key aspect for disrupting sex trafficking, not merely displacing it. Accounting for responses from traffickers is vital to *responsible* policy recommendations, as research suggests that removing individual victims from a trafficking situation, while clearly necessary, might paradoxically result in more victims being recruited into trafficking after an interdiction [27, 79]. This paradox aligns with the theoretical analysis in Chapter 3, where they observed that current law enforcement policy typically recommended interdicting participants in drug smuggling networks that would trigger restructuring arcs that would cross the minimum cut. We find that a similar situation occurs for our conceptualizations of sex trafficking networks, thus indicating that both disrupting current operations and their ability to restructure (especially recruit) is critial.

Recall the definition of MFNIP-R from Chapter 3: Given a network $G = (N, A)$ with source node α, sink node ω, set of restructurable arcs A^R, and node and arc capacities

$u : N \cup A \cup A^R \to \mathbb{R}_{\geq 0}$, let Y be the set of feasible interdiction plans and let $Z(y)$ be the set of feasible restructurings dependent on the chosen interdiction plan. First, the attacker chooses an interdiction plan $y \in Y$, setting the capacity of the interdicted nodes to 0. Next, the defender chooses a restructuring plan $z \in Z(y)$ to add arcs to the network, increasing their capacity from 0. Lastly, the defender operates the network to maximize flow through the network.

4.4.1 Modeling Sex Trafficking with Max Flow

Compared to drug trafficking, it is less clear in sex trafficking networks of traffickers and victims what an interdiction might seek to restrict in terms of flow. The function of a drug trafficking network is to move and eventually sell a product (i.e. drugs) to a user. Thus the flow is the drugs. In a sex trafficking network, flow is more complex. The "product" begin sold in sex trafficking is people and a sexual experience [80, 81]. The commercialization and sale of sex to a sex buyer may in some ways be analogous to the sale of drugs since both are commodities. The flow within the network is not the same because people (and their labor) are not equivalent to drugs. Much more work is needed to conceptualize how this concept of "flow" should best be applied to sex trafficking networks. Thus, we posit a novel interpretation of the maximum flow problem as it pertains to traffickers and victims of trafficking within a sex trafficking operation. Flow could represent the ways traffickers exhibit control over victims and the sex acts that victims are forced to perform. Thus flow is exerted through control over trafficking victims in order to make them (and their labor) a product to sell [81]. This is a preliminary exploration; much more empirical research is needed to verify and extend modeling to understand the specificity and nuances of trafficking for sexual exploitation. Further, it must be noted that the flow that is being modeled is exerted through violence, manipulation, and harm. For this discussion, we will focus on control within the network and on situations in which a trafficker needs to maintain control over a certain number of victims. We postulate, for the sake of this model, that a trafficker has a specific number of victims they wish to have under their control at any given time.

In the max flow network interdiction setting, we consider a model where nodes are interdicted, as opposed to arcs, so that interdictions are representative of removing traffickers and victims from the network. An equivalency between node interdiction and arc interdiction has been demonstrated in [75]. This relies on the standard network expansion technique of

replacing each node with two nodes and an arc between them, where all arcs coming into the original node flow into one of the expanded node, an arc goes to the second expanded node, and then all arcs leaving the original node leave the second expanded node. In this case, capacities are assigned to the nodes as well.

To convert an output of the network generator to a network usable in max flow network interdiction, we need to add in a source node and sink node. We add arcs from the source node to each trafficker and bottom node with infinite capacity, and arcs from each bottom and victim node to the sink with capacity one. All arcs incident to the trafficker are directed out of the trafficker, and all arcs incident to the bottom and a victim are directed out of the bottom and into the victim. Arcs between victims are replaced with a pair of arcs between the victims, one in each direction. The capacity of each trafficker node is the number of victims they can control, the capacity of each victim node is one, and the capacity of each bottom node is the number of victims they can control plus one. This is to recognize that bottoms are victims too, and that their role in sex trafficking networks does not make them any less of a victim than the other victims. Note that it is not necessary to directly model the control of a bottom with an arc from the trafficker to them as the arc from the source can model this control. For ease of notation, we define T to be the set of traffickers, B to be the set of bottoms, and V to be the set of victims. Capacities for the traffickers and bottoms are chosen in such a way that the maximum flow through the network is the total number of victims currently in the network (including bottoms). The arcs between traffickers are only used to determine restructurable arcs, and are assigned capacity 0.

We now define the set of restructurable arcs A^R. For the set of restructurable arcs, we allow for traffickers to recruit each other's victims, traffickers to give or take victims from their bottom, the recruitment of new victims, back-up traffickers to take over an interdicted trafficker's operation, the promotion of a new bottom, and traffickers to give victims to their newly promoted bottom. We start with A^R consisting of the set of arcs between traffickers and the victims of traffickers they know. More precisely, for $i \in T$ and $j \in V$, $(i, j) \in A^R$ if there exists a $k \in T$ such that $(i, k) \in A$ and $(k, j) \in A$. To allow for a trafficker i to assign a new victim k to their bottom j, we include $(j, k) \in A^R$ if $(j, k) \notin A$. Likewise, we allow for a trafficker i to take a victim k from their bottom j by including $(i, k) \in A^R$ if $(i, k) \notin A$.

To model the recruitment of new victims, we introduce an additional set of nodes V^R. We include an arc from each of these nodes to the sink node in the arc set A. We then

randomly determine a subset of traffickers that can restructure to each recruitable victim node, and add arcs between the traffickers and the recruitable victim to A^R. This results in these nodes having no arcs into them in the initial network, meaning no flow can pass through them, but flow can pass through them if an arc ending at the recruitable victim is restructured by a trafficker. To model an operation having a back-up trafficker, we introduce an additional set of nodes T^R to be the set of back-up traffickers for certain operations, which is represented by ordered pairs (i, j), where trafficker i can be replaced by back-up trafficker j if trafficker i is interdicted. We add arcs from the back-up trafficker j to all of trafficker i's victims, and include an arc (α, j) in the set of restructurable arcs. As with recruitable victims, there are no arcs into back-up trafficker j in the initial network, so no flow can pass through the node, but restructuring the arc (α, j) allows for flow to pass through the node. To model the promotion of new bottoms, we define the set B^R as the set of victims which may be eligible to be promoted to the role of bottom, which is represented by ordered pairs (i, j), where victim j may be promoted if bottom i was interdicted. If victim j is promoted to the role of bottom, their capacity is increased by $\tilde{u}_j$, allowing flow to pass from victim j to the victims that they arc adjacent to. An arc (α, j) is added to the set of restructurable arcs, mimicking the inclusion of arcs to back-up traffickers. We also allow for a trafficker i to assign a victim k to the newly promoted bottom j by including arc $(j, k) \in A^R$ if $(j, k) \notin A$ that can only be restructured if (α, j) has been restructured.

We now define the necessary components for MFNIP-R. Let x_{ij} be the amount of flow through arc (i, j), and x_i be the amount of flow through node i. Let y_i be the indicator of whether or not node i has been interdicted, setting its capacity to 0, with Y being the set of feasible interdiction decisions. As in Chapter 3, we define the variables on restructuring to represent which node is enabling the restructuring to occur. An "out" restructuring would be representative of a trafficker restructuring to a new victim after having one of their victims interdicted. An "in" restructuring would be representative of a victim being recruited into a new operation after their trafficker has been interdicted. Let z_{ij}^{out} be the indicator of restructuring arc (i, j) from node i, and let z_{ij}^{in} be the indicator of restructuring arc (i, j) from j, with $Z^{out}(y)$ and $Z^{in}(y)$ being the sets of feasible restructuring decisions from i or j, respectively, with respect to the interdiction decision y. We note that we restrict the set of restructuring decisions to be dependent on the implemented interdiction decisions because we are specifically interested in how the traffickers will react to the interdictions. Additionally,

we define sets of restructurable arcs $A^{R,out}$ and $A^{R,in}$, based on whether a restructurable arc in A^R is initiated by a trafficker or by a victim. restructurable arcs between an existing trafficker and existing victim are included in $A^{R,in}$, while all restructurable arcs are included in $A^{R,out}$. Note that $A^{R,in} \subset A^{R,out} = A^R$ in our model.

In this case study, instead of assigning a global budget for restructuring, we assign each trafficker a budget. We choose to limit the number of actions each trafficker can take, as the actions of one trafficker may not prevent the actions of the other traffickers. For ease of defining constraints, we further partition A^R into sets based on which trafficking operations are involved. Let $A_i^{R,out}$ be the set of restructurable "out" arcs that can be restructured by trafficker i, and $A_i^{R,in}$ be the set of restructurable "in" arcs that involve trafficker i. We note that these sets are disjoint between different traffickers, i.e. $A_i^{R,out} \cap A_j^{R,out} = \emptyset$ for $i, j \in T$ and $i \neq j$, and $A^{R,out} = \cup_{i \in T} A_i^{R,out}$.

Let c_{jk}^i be the cost for trafficker i to restructure the arc (j,k), and let b^i be the budget of trafficker i. For a fixed attacker solution $\bar{y}$, we can define the defender's problem as:

$$\max_{x,z} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha i} \tag{4.1a}$$

$$\text{s.t.} \sum_{(h,i) \in A \cup A^{R,out}} x_{hi} = x_i \text{ for } i \in N \tag{4.1b}$$

$$x_i = \sum_{(i,h) \in A \cup A^{R,out}} x_{ih} \text{ for } i \in N \tag{4.1c}$$

$$x_{ij} \leq u_{ij} \text{ for } (i,j) \in A \tag{4.1d}$$

$$x_{ij} \leq u_{ij} z_{ij}^{out} \text{ for } (i,j) \in A^{R,out} \setminus A^{R,in} \tag{4.1e}$$

$$x_{ij} \leq u_{ij}(z_{ij}^{out} + z_{ij}^{in}) \text{ for } (i,j) \in A^{R,in} \tag{4.1f}$$

$$x_i \leq u_i(1 - y_i) \text{ for } i \in N \setminus \{j \in V : \exists h \in B \text{ with } (h,j) \in B^R\} \tag{4.1g}$$

$$x_j \leq u_j(1 - y_j) + \tilde{u_j} z_{sj}^{out} \text{ for } j \in V \text{s.t.} \exists i \in B, (i,j) \in B^R \tag{4.1h}$$

$$\sum_{(j,k) \in A_i^{R,out}} c_{jk}^i z_{jk}^{out} + \sum_{(j,k) \in A_i^{R,in}} c_{jk}^i z_{jk}^{in} \leq b^i \text{ for } i \in T \tag{4.1i}$$

$$\sum_{j \in V} z_{ij}^{out} \leq \sum_{h \in V:(i,h) \in A} \bar{y}_h \text{ for } i \in T \tag{4.1j}$$

$$\sum_{i \in T} z_{ij}^{in} \leq \sum_{h \in T:(h,j) \in A} \bar{y}_h \text{ for } j \in V \tag{4.1k}$$

$$\sum_{(i,j) \in T^R} z_{\alpha j}^{out} \leq \bar{y}_i \text{ for } i \in T \tag{4.1l}$$

$$\sum_{(i,j) \in B^R} z_{\alpha j}^{out} \leq \bar{y}_i \text{ for } i \in B \tag{4.1m}$$

$$z_{\alpha j}^{out} \leq 1 - \bar{y}_j \text{ for } j \in V \text{ s.t. } \exists i \in B, (i,j) \in B^R \tag{4.1n}$$

$$\sum_{(j,k)\in B^R:(i,j)\in A} z_{sk}^{out} \leq 1 \text{ for } i \in T \tag{4.1o}$$

$$\sum_{k\in V:(j,k)\in A_i^{R,out}} z_{jk}^{out} \leq |\{k \in V : (j,k) \in A_i^{R,out}\}| z_{\alpha j}^{out} \text{ for } i \in T, j \in V \text{ s.t. } \exists h \in B, (h,j) \in B^R \tag{4.1p}$$

$$\sum_{i\in T:(i,j)\in A_i^{R,out}\cap A_i^{R,in}} (z_{ij}^{in} + z_{ij}^{out}) \leq 1 \text{ for } j \in V \tag{4.1q}$$

$$(z_{ij}^{out} + z_{ij}^{in}) \leq 1 \text{ for } (i,j) \in A_i^{R,out} \cap A_i^{R,in} \tag{4.1r}$$

$$x \geq 0 \tag{4.1s}$$

$$z^{out} \in \{0,1\}^{|A^{R,out}|} \tag{4.1t}$$

$$z^{out} \in \{0,1\}^{|A^{R,in}|} \tag{4.1u}$$

The objective function of the MILP is maximizing the amount of flow out of the source node. Constraints (4.1b)-(4.1c) are flow balance constraints. Constraints (4.1d)-(4.1h) enforce the flow through an arc or node is at most the capacity of that arc or node, as adjusted by interdictions and restructurings. In particular, constraints (4.1h) incorporates the decrease in flow through a victim node from interdiction and increase from being promoted to the role of bottom. Constraints (4.1i) enforces the restructuring budget for each trafficker. Constraints (4.1j)-(4.1k) allow for restructurings from the traffickers or bottoms based on the implemented interdictions. Constraints (4.1l) allow for a back-up trafficker to be restructured to if the trafficker has been interdicted, and constraints (4.1m) allow for a victim to be promoted to bottom if the operation's bottom has been interdicted. Constraint (4.1n) prevents an interdicted victim from being promoted to bottom, and constraint (4.1o) enforces that at most one victim can be promoted to bottom. Constraint (4.1p) allows for the trafficker to give victims to the newly promoted bottom. Constraint (4.1q) prevents a victim from being recruited into more than one operation. For arcs in $A_i^{R,out} \cap A_i^{R,in}$, constraint (4.1r) prevents both z^{out} and z^{in} from being nonzero. Constraint (4.1s) enforces that the flow variables are non-negative, and constraints (4.1t)-(4.1u) enforces that the restructuring variables are binary.

The attacker is assigned a budget b^a to interdict the network, limiting their ability to disrupt the network. Each node i is assigned a cost r_i that must be expended to interdict that node. In drug trafficking, "climbing the ladder" constraints are used to emulate how law enforcement would pursue prosecuting cases against participants in the drug trafficking network [75]. In disrupting sex trafficking networks, it is not necessary to interdict victims in order to interdict traffickers; however the cooperation of victims can assist in the successful

prosecution of traffickers. To model this, the cost of interdicting a trafficker node is reduced by a fixed amount based on whether or not their bottom has been interdicted, as well as the number of their victims that have been interdicted. Let d_k^i be the decrease in interdiction cost of trafficker i by interdicting node k. Additionally, let r_i^{min} be the minimum cost to interdict trafficker i, and define the variable $\tilde{r}_i$ be the adjusted cost of interdicting trafficker i.

We can define the attacker's problem as:

$$\min_{y,\tilde{r}} \max_{x,z} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha i} \tag{4.2a}$$

$$\text{s.t.} \quad \text{Constraints (4.1b) - (4.1u)} \tag{4.2b}$$

$$\sum_{i \in T} \tilde{r}_i y_i + \sum_{j \in B \cup T} r_i y_i \leq b^a \tag{4.2c}$$

$$\tilde{r}_i \geq r_i^{min} \text{ for } i \in T \tag{4.2d}$$

$$\tilde{r}_i \geq r_i - \sum_{k \in B \cup V} d_k^i y_i \text{ for } i \in T \tag{4.2e}$$

$$y \in \{0,1\}^{|N|} \tag{4.2f}$$

Constraint (4.2c) is the budget of the attacker. Constraints (4.2d) and (4.2e) adjust the cost to interdict the traffickers. Constraint (4.2f) enforces that the interdiction variables are binary. We note that, as qualitative research on the operations of sex trafficking networks is furthered, this model can be refined to provide more accurate insights regarding disrupting sex trafficking networks.

Note that $\tilde{r}_i y_i$ is a bilinear term. This can be linearized by replacing it with a new variable $\tilde{y}_i$ and including the McCormick inequalities in the set of constraints [84]. Since y_i is binary, the McCormick inequalities will result in $\tilde{y}_i = \tilde{r}_i y_i$.

We apply the method derived in Chapter 3 to solve this model. Recall that this method is a column-and-constraint generation (C&CG) method, where, when a previously visited restructuring plan is infeasible with respect to the currently interdiction plan, the feasible components in that restructuring plan are used in place of the full plan to determine a lower bound on the true objective value. This method has been shown to be effective in solving their model faster than standard C&CG methods, particularly when new participants are able to be recruited.

4.4.2 Computational Results

We implement this interdiction model on five generated networks, each with five operations. Table 4.3 reports the number of nodes, and number of bottoms and victims, in each network.

Table 4.3: Sizes of generated networks

Network	Number of Nodes	Number of Traffickers	Number of Bottoms	Number of Victims
1	28	5	3	20
2	32	5	4	23
3	35	5	5	25
4	35	5	4	26
5	27	5	4	18

We set the cost of interdicting a trafficker to be 8, the cost to interdict a bottom to be 4, and the cost to interdict a victim to be 2. Interdicting a bottom reduces the cost of interdicting their trafficker by 3, and interdicting a victim reduces the cost of interdicting their trafficker by 1. Each trafficker has a budget of 8 to restructure their operation. The cost to restructure to a victim currently in the network is 1, as is the cost to take a victim from their bottom or give a victim to their bottom. The cost to recruit a new victim not currently in the network is 2. The cost for a back-up trafficker to come into the operation is 4. The cost to promote a new bottom is 5, and the cost for the trafficker to give them a victim is 2. We set the number of recruitable victims to be 40% of the number of victims in the network. A trafficker will have a back-up trafficker if they have at least four victims (including a bottom). If a trafficker has a bottom, the number of victims that can be promoted to become a bottom is half of the number of victims the trafficker is connected to (minimum one).

We set a solve time limit of 2 hours. All instances solved within the time limit. We model the problem in AMPL with Gurobi 9.0.2 as the solver. Experiments were conducted on a laptop with an Intel® Core™ i5-8250 CPU @ 1.6 GHz - 1.8 GHz and 16 GB RAM running Windows 10. Figure 4.5 demonstrates the flow through the network over different attacker budgets. In this figure, the dotted black represents the total number of victims (including bottoms) in the network. The hollow circles represent the interdicted flow determined by solving MFNIP. The stars represent the flow after optimally restructuring the network in response to MFNIP's recommended interdictions. The diamonds represent the interdicted flow determined by MFNIP-R. Note that network 1 is the network in Figure 4.4.

In each network and across all budget levels, the traffickers are able to recover a significant amount of flow after restructuring in response to the MFNIP recommended interdictions. This is primarily due to being able to recruit new victims when victims are primarily interdicted, and being able to replace interdicted traffickers with back-up traffickers when traffickers are primarily interdicted. However, when we are able to account for how the traffickers will restructure in response to interdiction, we are often able to reduce the flow. Table 4.4 demonstrates the interdiction recommendations for the network in Figure 4.4. Results on other networks can be found in Appendix E. Columns 2 - 4 display the number of traffickers, bottoms, and victims that MFNIP recommends to interdict, and columns 5 - 7 display the number of traffickers, bottoms, and victims that MFNIP-R recommends to interdict.

Table 4.4: Interdiction recommendations from MFNIP and MFNIP-R for network 1

Attacker Budget	MFNIP Int Trafficker	MFNIP Int Bottom	MFNIP Int Victim	MFNIP-R Int Trafficker	MFNIP-R Int Bottom	MFNIP-R Int Victim
8	0	2	0	0	0	4
12	1	1	0	1	0	4
16	1	2	1	1	0	6
20	1	2	3	1	1	6
24	1	3	3	1	0	10
28	1	3	4	1	2	8
32	1	3	8	2	1	8
36	1	3	10	2	1	12
40	1	3	12	2	3	10

A clear distinction between the two recommended plans is the number of bottoms that each model recommends to interdict. Consistently across all attacker budgets, MFNIP recommends interdicting at least as many bottoms than MFNIP-R does. This observation is consistent across all tested networks. This is likely due to MFNIP-R identifying that bottoms have limited ability to restructure the network, as the only way they can acquire new victims is if their trafficker assigns them new victims. This seems to highlight that limiting the ability of the trafficker to restructure after disruptions may be just as important as disrupting the current operations, which is similar to applying the theoretical analysis of Chapter 3 to the MFNIP-R applied to sex trafficking networks. Additionally, network 3, where every operation has a bottom, had the smallest gap in flows projected by MFNIP and MFNIP-R. These observations suggest that better understanding how bottoms help operate

the network both before and after disruptions may be critical to understanding how to more effectively dismantle sex trafficking networks.

4.5 Limitations and Directions for Future Qualitative Research

As the lack of data is the challenge we wish to address, our network generator has many limitations that a responsible user needs to be aware of. First and foremost, our network generator is built on sources about domestic sex trafficking cases within the Midwest of the USA, since the case file analysis focused on the cases in the Midwest and the advisory group's experiences involve operations within the Midwest. As such, their experience is not necessarily representative of sex trafficking as a whole. It may be that biases based on their experience are built into the network generator. However, the more the network generator is used and feedback is provided regarding operations in more locations and contexts, the more accurate we will be able to make it over time. Users should collaborate with domain experts in their geographical location to understand how sex trafficking networks in their location may operate differently from that of the Midwest.

Another limitation is regarding the social connections between traffickers. It is known that some traffickers are connected socially with other traffickers, but there has been little research on what these social networks look like. This gap prevents us from developing more authentic trafficker social networks. Further research focused on how traffickers are connected, both socially and professionally, will allow for more accurate trafficker social networks to be produced.

How bottoms interact with the network is also an area where further research is needed. While their function in the network is known, little is known on how they are connected to other traffickers and victims outside of their operation. It is also not well known what causes a trafficker to want or need to have a bottom in their network. Here, we base the likelihood of there being a bottom on the number of victims in the trafficker's operations. Future qualitative research can ascertain why a trafficker chooses to promote a victim to the role of the bottom and how having a bottom allows a trafficker to grow their business.

4.6 Conclusions

We presented our network generator for producing synthetic domestic sex trafficking networks. The networks produced include the operational and social connections between

traffickers, bottoms, and victims. Data sources including publicly available federal case files and interviews were triangulated to determine the basis and parameters in the generator. The generator was further validated by domain experts, including a survivor-centered advisory group. As qualitative research furthers our understanding of the connections within sex trafficking networks, we can refine and advance the functionality of the generator in future iterations. An example of such research is on how traffickers communicate and collaborate, which would allow us to build multiple-trafficker operations. The network generator and generated networks in this study are available from the corresponding author upon reasonable request.

This generator allows for the OR community to engage in human trafficking research without needing to go through the time-intensive process of collecting and cleaning their own data. We demonstrate how an OR tool, network interdiction, can be applied to networks produced by this generator. In particular, we proposed a novel conceptualization of flow which considers the ability of the traffickers to control their victims. Our results suggest that better understanding the roles of bottoms in maintaining operations after disruptions occur may be key to more impactful disruptions.

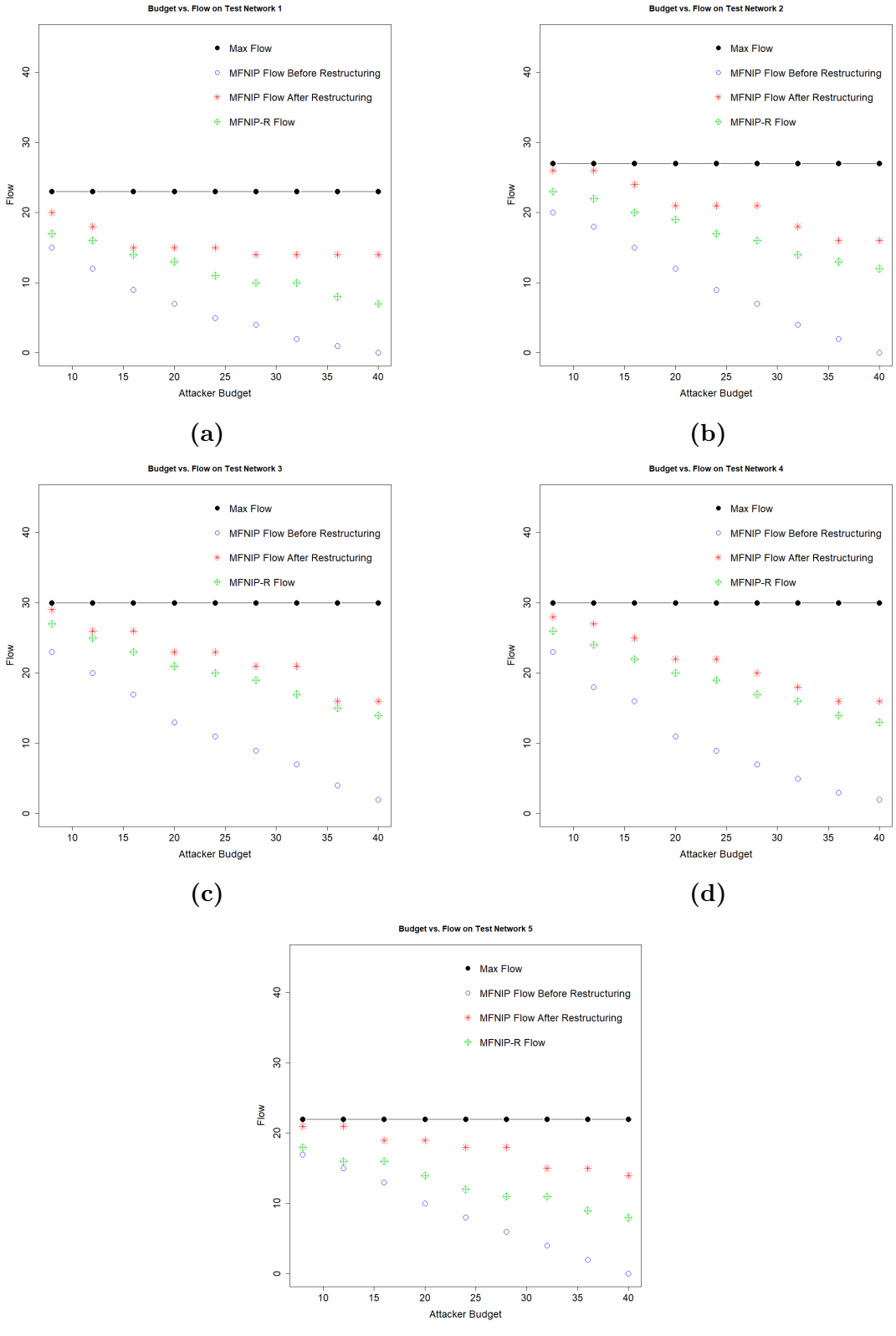


Figure 4.5: MFNIP-R flows over varying attacker budgets on (a) network 1, (b) network 2, (c) network (3), (d) network 4, and (e) network 5

CHAPTER 5
MULTI-PERIOD MAX FLOW NETWORK INTERDICTION WITH RESTRUCTURING FOR DISRUPTING DOMESTIC SEX TRAFFICKING NETWORKS

5.1 Introduction

In this chapter, we extend max flow network interdiction with restructuring (MFNIP-R) to include a temporal component. We refer to this problem as multi-period max flow network interdiction with restructuring (MP-MFNIP-R). Modeling interdictions and restructurings for domestic sex trafficking networks is extended to include a temporal component. We also introduce a variant of MP-MFNIP-R to include a second cooperating attacker that also has the ability interdict victims and to prevent prospective victims from being recruited, modeling the how social services would disrupt the network. We propose two augmentations to the column-and-constraint generation procedure to improve the solve time of the method.

This chapter is organized as follows: Section 5.2 introduces the multi-period max flow network interdiction problem with restructuring, and models the problem as a bilevel mixed integer-linear program (BMILP). Section 5.3 discusses how domestic sex trafficking networks can be modeled with a max flow model and proposes constraints on the interdiction. Section 5.4 discusses the constraints on restructuring based on the set of restructureable arcs proposed in Chapter 4. Section 5.5 derives a column-and-constraint generation(C&CG) algorithm to solve the BMILP, and demonstrates equivalent models based on various modeling choices. Section 5.6 proposes modeling-based augmentations to the C&CG algorithm. Section 5.7 presents computational results on the solve time of the various models and solution methods, as well as discusses policy analysis of recommended interdictions. Section 5.8 concludes the chapter.

5.2 Problem Description

We now introduce the multi-period max flow network interdiction problem with restructuring (MP-MFNIP-R). MP-MFNIP-R is a two player game on a network $G = (N, A)$, with N being the set of nodes and A being the set of arcs. In applying max flow interdiction to disrupting domestic sex trafficking networks, we specify that the set of interdiction

decisions is restricted to the set of nodes, and restructuring decisions are represented by $A^R = A^{R,out} \cup A^{R,in}$, as described in Chapter 3. Nodes and arcs (including restructurable arcs) are assigned capacities $u : N \cup A \cup A^R \to \mathbb{R}_+$. We denote that $\alpha \in N$ is the source node, and $\omega \in N$ is the sink node. Let τ be the number of time periods.

Gameplay for MP-MFNIP-R is as follows. First, the attacker decides upon an interdiction plan. Interdictions are decided upon at the beginning of the time horizon, and require a certain length of time before they are implemented, setting the capacity of the interdicted node to 0. After the interdiction plan is decided upon, the defender decides upon a restructuring plan in response to the attacker's interdiction plan. Restructuring decisions are also decided upon at the beginning of the time horizon, and require a certain length of time before they are implemented, setting the capacity of each restructured arc to its non-zero capacity. After the interdiction and restructuring decisions are made, the defender operates the network, sending flow from source to sink instantaneously each time period. The goal of the defender is to maximize the amount of flow sent throughout the entire time horizon, and the goal of the attacker is to minimize the amount of flow sent throughout the entire time horizon.

Recall from Chapter 4 that T is the set of traffickers, B is the set of bottoms, and V is the set of victims. Additionally, T^R is the set of back-up traffickers, B^R is the set of victims that can be promoted to be a bottom, and V^R is the set of prospective victims.

To describe the mathematical program, we must first define the decision variables and relevant parameters. Let

$$y_i = \begin{cases} 1 & \text{if node } i \text{ has been interdicted,} \\ 0 & \text{otherwise,} \end{cases} \tag{5.1}$$

and let

$$\gamma_{it} = \begin{cases} 1 & \text{if node } i \text{ is interdicted in or before time period } t, \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

Let δ_i^y be the number of time periods needed before node i can be interdicted. Additionally, let

$$z_{ij}^{out} = \begin{cases} 1 & \text{if arc } (i,j) \text{ has been restructured when } i \text{ is able to initiate the restructuring,} \\ 0 & \text{otherwise,} \end{cases} \tag{5.3}$$

and

$$z_{ij}^{in} = \begin{cases} 1 & \text{if arc } (i,j) \text{ has been restructured when } j \text{ is able to initiate the restructuring,} \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

Let

$$\zeta_{ijt}^{out} = \begin{cases} 1 & \text{if arc } (i,j) \text{ has been restructured in or before time period } t \\ & \text{when } i \text{ is able to initiate the restructuring,} \\ 0 & \text{otherwise,} \end{cases} \tag{5.5}$$

and

$$\zeta_{ijt}^{in} = \begin{cases} 1 & \text{if arc } (i,j) \text{ has been restructured in or before time period } t \\ & \text{when } j \text{ is able to initiate the restructuring,} \\ 0 & \text{otherwise.} \end{cases} \tag{5.6}$$

Let δ_{ij}^{z} be the number of time periods needed before arc (i,j) can be restructured. We can now describe the bilevel programming formulation of MP-MFNIP-R.

$$\min_{y,\tilde{r}} \max_{x,z,\zeta} \sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha it} \tag{5.7a}$$

$$\text{s.t.} \sum_{(h,i) \in A \cup A^{R,out}} x_{hit} = x_{it} \text{ for } i \in N \setminus \alpha, t = 1, \ldots, \tau \tag{5.7b}$$

$$x_{it} = \sum_{(i,h) \in A \cup A^{R,out}} x_{iht} \text{ for } i \in N \setminus \omega, t = 1, \ldots, \tau \tag{5.7c}$$

$$0 \leq x_{ijt} \leq u_{ij} textfor (i,j) \in A, t = 1, \ldots, \tau \tag{5.7d}$$

$$0 \leq x_{ijt} \leq u_{ij} \zeta_{ij}^{out} \text{ for } (i,j) \in A^{R,out} \setminus A^{R,in}, t = 1, \ldots, \tau \tag{5.7e}$$

$$0 \leq x_{ijt} \leq u_{ij} (\zeta_{ijt}^{out} + \zeta_{ijt}^{in}) \text{ for } (i,j) \in A^{R,in}, t = 1, \ldots, \tau \tag{5.7f}$$

$$0 \leq x_{it} \leq u_i (1 - \gamma_{it}) \text{ for } i \in N \setminus \{j \in V : \exists h \in B \text{ with } (h,j) \in B^R\}, t = 1, \ldots, \tau \tag{5.7g}$$

$$0 \leq x_{jt} \leq u_j (1 - \gamma_{jt}) + \tilde{u}_j \zeta_{\alpha jt}^{out} \text{ for } j \in V \text{s.t.} \exists i \in B, (i,j) \in B^R, t = 1, \ldots, \tau \tag{5.7h}$$

$$(y, \gamma) \in Y \tag{5.7i}$$

$$(z, \zeta) \in Z(y) \tag{5.7j}$$

It is worth noting that this model builds off the work done Chapter 3 by adding the time dimension. We now describe how we model interdiction decisions and constraints on them.

5.3 Modeling Interdictions

We now describe constraints regarding interdiction, based on interdicting domestic sex trafficking networks. We first describe constraints linking y and γ.

$$\gamma_{it} = 0 \text{ for } i \in N \setminus \{\alpha, \omega\}, t \in 1, \ldots, \delta_i^y \tag{5.8}$$

$$\gamma_{it} = y_i \text{ for } i \in N \setminus \{\alpha, \omega\}, t \in (\delta_i^y + 1), \ldots, \tau \tag{5.9}$$

Constraints (5.8) indicate that a node i will be online for the first δ_i^y time periods, regardless of interdiction decisions. Then, after δ_i^y time periods have elapsed, constraints (5.9) enforce that the node will be offline if it was interdicted.

We additionally include a budget constraint to limit the overall number of interdictions implemented. We use the same budget constraints described in Chapter 4, where the cost to interdict a trafficker can be decreased based on interdicting their bottom (if the trafficker has one) and victims. To represent this, we define additional variables $\tilde{r}_i$ be the adjusted cost of interdicting trafficker i. Let r_i be the cost to interdict node i and let b_a be the total budget of the attacker. Additionally, for each trafficker $i \in T$, let r_i^{min} be the minimum cost of interdicting trafficker i after interdicting their victims, and let d_{il} be the reduction in cost of interdicting trafficker i if victim l is also interdicted. The following constraints capture the interdiction budget and costs to interdict each node.

$$\sum_{i \in T} \tilde{r}_i y_i + \sum_{j \in B \cup N} r_i y_i \leq b_a \tag{5.10}$$

$$\tilde{r}_i \geq r_i - \sum_{l \in B \cup V} d_{il} y_i \text{ for } i \in T \tag{5.11}$$

$$\tilde{r}_i \geq r_i^{min} \text{ for } i \in T \tag{5.12}$$

Constraint (5.10) enforces that the chosen interdictions respect the overall attacker budget. Constraints (5.11)-(5.12) compute the adjusted cost to interdict a trafficker.

5.3.1 Modeling Adaptation: Cooperating Attackers

There are multiple stakeholders in the efforts to disrupt human trafficking, which have a variety of different means of disrupting the network. For example, law enforcement has the ability to arrest and prosecute participants in the trafficking network, while social service professionals have the ability to provide services to victims to help them leave the trafficking network while also helping reduce the vulnerabilities of prospective victims.

We want to model a second interdictor that only has the ability to interdict victims and prevent the ability to add prospective victims to the network. Let y'_i be the indicator of whether or not the second attacker interdicts node i for $i \in V \cup V^R$. Let c'_i be the cost for the second interdictor to interdict node i, and let b' be the budget of the second attacker. We include the following constraint regarding the second attacker:

$$\sum_{i \in V \cup V^R} c'_i y'_i \leq b' \tag{5.13}$$

To update the model to incorporate the second attacker, we must update the existing constraints regarding interdicting victims, as well as include constraint (5.13). For $i \in V$, we adjust the constraints (5.9) from $\gamma_{it} = y_i$ to $\gamma_{it} = y_i + y'_i$ for $t \in (\delta^y_i + 1), \ldots, \tau$. For $i \in V^R$, we introduce the constraints $\gamma_{it} = y'_i$ for $t = 1, \ldots, \tau$. If a prospective victim j were to be interdicted, then even if an arc (i, j) were to be restructured from trafficker i, the capacity of the prospective victim node would be set to 0 for $t = \delta^y_j, \ldots, \tau$, and thus would not increase the objective value in time periods after the victim is interdicted.

5.4 Modeling Restructuring

For the set of restructurable arcs, we use the same sets outlined in Chapter 4. We extend the constraints on the restructuring outlined in Chapter 4 to a multi-period setting. We first propose constraints that link when interdictions occur to when restructurings are allowed. For $i \in T$, let $\delta_i^{min,out} = \min_{(i,j) \in A^{R,out}} \delta^z_{ij}$, and $\delta_i^{max,out} = \max_{(i,j) \in A^{R,out}} \delta^z_{ij}$, which help to capture the earliest and the latest time which a trafficker may initiate a restructuring after an interdiction has occurred. For $i \in V$, let $\delta_i^{min,in} = \min_{(i,j) \in A^{R,in}} \delta^z_{ij}$, and $\delta_i^{max,in} = \max_{(i,j) \in A^{R,in}} \delta^z_{ij}$, which help to capture the earliest and the latest time which a victim may initiate a restructuring after an interdiction has occurred.

$$\sum_{(i,j)\in A_R^{out}} z_{ij}^{out} \leq \sum_{h\in N:(i,h)\in A} \bar{y}_h \text{ for all } i \in T \tag{5.14}$$

$$\sum_{(i,j)\in A_R^{in}} z_{ij\tau}^{in} \leq \sum_{h\in N:(h,j)\in A} \bar{y}_h \text{ for all } j \in V \tag{5.15}$$

$$\sum_{\substack{j\in N:(i,j)\in A^{R,out} \\ \delta_{ij}^z \geq t_1}} \zeta_{ij(t_2+\delta_{ij}^z)}^{out} \leq \sum_{h:(i,h)\in A,(\delta_h^y+1)\leq t_2} \bar{y}_h \tag{5.16}$$

$$\text{for } i \in T, t_1 = \delta_i^{min,out}, \ldots, \delta_i^{max,out}, t_2 = 1, \ldots, \tau - \delta_i^{max,out}$$

$$\sum_{\substack{j\in N:(j,i)\in A^{R,in} \\ \delta_{ji}^z \geq t_1}} \zeta_{ji(t+\delta_{ji}^z)}^{in} \leq \sum_{h:(h,i)\in A,(\delta_h^y+1)\leq t} \bar{y}_h \tag{5.17}$$

$$\text{for } i \in T, t_1 = \delta_i^{min,in}, \ldots, \delta_i^{max,in}, t_2 = 1, \ldots, \tau - \delta_i^{max,in}$$

Constraints (5.14)-(5.15) determine the number of restructurings each trafficker and victim is allowed to make in response to the implemented interdictions. Constraints (5.16)-(5.17) enforce that a restructured arc can only come online after the required amount of time has passed. We next define constraints to enforce the relationship between z and ζ.

$$\zeta_{ijt}^{out} \geq \zeta_{ij(t-1)}^{out} \text{ for } (i,j) \in A^{R,out}, t = 2, \ldots \tau \tag{5.18}$$

$$\zeta_{ijt}^{in} \geq \zeta_{ij(t-1)}^{in} \text{ for } (i,j) \in A^{R,in}, t = 2, \ldots \tau \tag{5.19}$$

$$\zeta_{ij\tau}^{out} = z_{ij}^{out} \text{ for } (i,j) \in A^{R,out} \tag{5.20}$$

$$\zeta_{ij\tau}^{in} = z_{ij}^{out} \text{ for } (i,j) \in A^{R,in} \tag{5.21}$$

Constraints (5.18)-(5.19) enforce that if an arc has turned online, it must remain online in the following time period. Constraints (5.20)-(5.21) ensure that if an arc was restructured, it will be online in the last time period. The combination of these constraints, as well as (5.14)-(5.17), ensure that, once it is decided that an arc is to be restructured, it will first come online in a time period that it is allowed to, and remain online for the rest of the time horizon. They also ensure that the restructurings chosen occur in the appropriate time periods based on the chosen interdictions, and that the number of restructurings is limited by the number of relevant interdictions.

The next constraints limit the number of actions that traffickers can take over the

entire time horizon, independent of the number of interdictions that have occurred. We note that the constraints include the ability of each trafficker to bring new victims into their organization, assign new victims to their bottom, and to promote a victim to be a bottom.

$$\sum_{j\in N:(i,j)\in A^{R,out}} z_{ij}^{out} + \sum_{\substack{k\in N,h\in N:(i,k)\in A \\ (k,h)\in A^{R,out}}} z_{kh}^{out} + \sum_{\substack{l\in N:(i,l)\in A \\ (\alpha,l)\in A^{R,out}}} z_{\alpha l}^{out} \leq c_i^{out} \text{ for all } i \in T \tag{5.22}$$

We additionally include constraints that limit the number of interactions with each victim (including prospective victims), preventing them from belonging to too many trafficking operations.

$$\sum_{(i,j)\in A^{R,out}} z_{ij}^{out} + \sum_{(i,j)\in A^{R,in}} z_{ij}^{in} \leq c_j^{in} \text{ for } j \in V \cup V^R \tag{5.23}$$

For trafficking operations that have a back-up trafficker, we include constraints indicating when a back-up trafficker can replace an interdicted trafficker.

$$\zeta_{\alpha jt}^{out} = 0 \text{ for } (i,j) \in T^R, t \in 1,\ldots,(\delta_i^y + \delta_{\alpha j}^{z,out}) \tag{5.24}$$

$$\zeta_{\alpha jt}^{out} = \bar{y}_i \text{ for } (i,j) \in T^R, t = (\delta_i^y + \delta_{\alpha j}^{z,out} + 1),\ldots,\tau \tag{5.25}$$

Constraints (5.24) enforce that the arc representing the back-up trafficker will be offline for a number of time periods equal to the time to interdict the primary trafficker and restructure to the back-up trafficker. Constraints (5.25) allow for the arc to come online if the trafficker has been interdicted. The next set of constraints enforces when victims are allowed to be promoted to be a bottom.

$$\zeta_{\alpha jt}^{out} = 0 \text{ for } (i,j) \in B^R, t = 1,\ldots,(\delta_i^y + \delta_{\alpha j}^{z,out}) \tag{5.26}$$

$$\sum_{(i,j)\in B^R} \zeta_{\alpha j(t+\delta_{\alpha j}^{z,out})}^{out} \leq \bar{y}_i \text{ for } i \in B, t = (\delta_i^y + 1),\ldots,\tau \tag{5.27}$$

$$z_{\alpha jt}^{out} \leq 1 - \bar{y}_j \text{ for } j \in V \text{ s.t. } \exists i \in B, (i,j) \in B^R \tag{5.28}$$

$$\sum_{h\in V:(j,h)\in A^{R,out}} z_{jh}^{out} \leq |V| z_{\alpha j}^{out} \text{ for } j \in V \text{ s.t. } \exists i \in B, (i,j) \in B^R \tag{5.29}$$

Constraints (5.26)-(5.27) act similarly to constraints (5.24)-(5.25), allowing a victim to be promoted to a bottom if the current bottom of the operations is interdicted and after the requisite time has passed. Constraints (5.28) prevent an interdicted victim from being promoted to be a bottom. Constraints (5.29) allows for a trafficker to assign some of their victims to the newly promoted bottom once they have been promoted.

The last constraints enforce that any arc in $A^{R,out} \cap A^{R,in}$ cannot have both z^{out} and z^{in} be nonzero.

$$z_{ij}^{out} + z_{ij}^{in} \leq 1 \text{ for all } (i,j) \in A^{R,out} \cap A^{R,in} \tag{5.30}$$

5.5 Model Derivation

We now describe the integer programming formulation of the multi-period max flow network interdiction problem with restructuring (MP-MFNIP-R) including the constraints on interdictions, and derive the column-and-constraint algorithm to solve it.

$$\min_{y,\tilde{r}} \max_{x,z,\zeta} \sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha it} \tag{5.31a}$$

$$\text{s.t.} \sum_{(h,i) \in A \cup A^{R,out}} x_{hit} = x_{it} \text{ for } i \in N \setminus \alpha, t = 1, \ldots, \tau \tag{5.31b}$$

$$x_{it} = \sum_{(i,h) \in A \cup A^{R,out}} x_{iht} \text{ for } i \in N \setminus \omega, t = 1, \ldots, \tau \tag{5.31c}$$

$$0 \leq x_{ijt} \leq u_{ij} \text{ for } (i,j) \in A, t = 1, \ldots, \tau \tag{5.31d}$$

$$0 \leq x_{ijt} \leq u_{ij} \zeta_{ij}^{out} \text{ for } (i,j) \in A^{R,out} \setminus A^{R,in}, t = 1, \ldots, \tau \tag{5.31e}$$

$$0 \leq x_{ijt} \leq u_{ij} (\zeta_{ijt}^{out} + \zeta_{ijt}^{in}) \text{ for } (i,j) \in A^{R,in}, t = 1, \ldots, \tau \tag{5.31f}$$

$$0 \leq x_{it} \leq u_i (1 - \gamma_{it}) \text{ for } i \in N \setminus \{j \in V : \exists h \in B \text{ with } (h,j) \in B^R\}, t = 1, \ldots, \tau \tag{5.31g}$$

$$0 \leq x_{jt} \leq u_j (1 - \gamma_{jt}) + \tilde{u_j} \zeta_{\alpha jt}^{out} \text{ for } j \in V \text{s.t.} \exists i \in B, (i,j) \in B^R, t = 1, \ldots, \tau \tag{5.31h}$$

$$\sum_{i \in T} \tilde{r}_i y_i + \sum_{j \in B \cup T} r_i y_i \leq b^a \tag{5.31i}$$

$$\tilde{r}_i \geq r_i^{min} \text{ for } i \in T \tag{5.31j}$$

$$\tilde{r}_i \geq r_i - \sum_{k \in B \cup V} d_k^i y_i \text{ for } i \in T \tag{5.31k}$$

$$\gamma_{it} = 0 \text{ for } i \in N, t = 1, \ldots, \delta_i^y \tag{5.31l}$$

$$\gamma_{it} = y_i \text{ for } i \in N, t \in (\delta_i^y + 1), \ldots, \tau \tag{5.31m}$$

$$y, \in \{0,1\}^{|N| \setminus \{\alpha,\omega\}} \tag{5.31n}$$

$$\gamma \in \{0,1\}^{|N| \setminus \{\alpha,\omega\} \times \tau} \tag{5.31o}$$

$$z, \zeta \in Z(y) \tag{5.31p}$$

We follow the procedure outlined in Chapter 3 to derive a single-level minimization problem that can be solved as part of column-and-constraint generation with partial information. For the sake of brevity, we include the full model derivation in Appendix F, and only state the final master problem that is solved.

We first describe variables associated with the standard column-and-constraint generation procedure. Let η be the variable representative of the objective value of the bilevel optimization problem, and let n be the number of restructurings plans being considered in the optimization model. Let $\pi_{it}^{+^k}$ and $\pi_{it}^{-^k}$ be the dual variables associated with constraints (5.31b) and (5.31c), respectively, for node i at time t for the k^{th} restructuring plan. Let θ_{ijt}^k be the dual variables associated with constraints (5.31d)-(5.31f) for arc (i, j) at time t for the k^{th} restructuring plan, and let θ_{it}^k be the dual variable associated with constraints (5.31g)-(5.31h) for node i at time t for the k^{th} restructuring plan.

Now we describe variables specific to the partial information adaptation. Let $w_{ijt}^{out,k}$ be the indicator of if arc (i, j) can be restructured out of node i in time period t for restructuring plan k, and let $w_{ijt}^{in,k}$ be the indicator of if arc (i, j) can be restructured in from node j in time period t for restructuring plan k. Let $\lambda_{it}^{out,k}$ be the indicator of if all of the "out" restructurings for trafficker i in time period t for restructuring plan k have been performed, and let $\lambda_{it}^{in,k}$ be the indicator of if all of the "in" restructurings for victim i in time period t for restructuring plan k have been performed.

$$\min_{y,\gamma,w\pi,\theta} \eta \tag{5.32a}$$

$$\text{s.t. } \eta \geq \sum_{t=1}^{\tau} [\sum_{i \in N \setminus \{\alpha,\omega\}} u_i \theta_{it}^k + \sum_{(i,j) \in A \cup A^{R,out} \cup A^{R,in}} u_{ij} \theta_{ijt}^k + \sum_{(i,j) \in B^R} \tilde{u_j} z_{\alpha jt}^{out,k} w_{\alpha jt}^{out,k} \theta_{\alpha jt}^k] \tag{5.32b}$$

$$\text{for } k = 1, \ldots, n$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq 1 \text{ for } k = 1, \ldots, n, (\alpha, j) \in A, t = 1, \ldots, \tau \tag{5.32c}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq 0 \text{ for } k = 1, \ldots, n, (i, j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau \tag{5.32d}$$

$$\pi_{it}^{-^k} - \pi_{it}^{+^k} + \theta_{it}^k \geq -\gamma_{it} \text{ for } k = 1, \ldots, n, i \in N \setminus \{\alpha, \omega\}, t = 1, \ldots, \tau \tag{5.32e}$$

$$-\pi_{it}^{-^k} + \theta_{i\omega t}^k \geq 0 \text{ for } k = 1, \ldots, n, (i, \omega) \in A, t = 1, \ldots, \tau \tag{5.32f}$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq w_{\alpha jt}^{out,k} + z_{\alpha j}^{out,k} - 1 \text{ for } k = 1, \ldots, n, (\alpha, j) \in A^{R,out}, t = 1, \ldots, \tau \tag{5.32g}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq w_{ijt}^{out,k} + z_{ij}^{out,k} - 2 \tag{5.32h}$$

$$\text{for } k = 1, \ldots, n, (i, j) \in A^{R,out} \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq w_{ijt}^{in,k} + z_{ij}^{in,k} - 2 \text{ for } k = 1, \ldots, n, (i,j) \in A^{R,in} \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau \quad (5.32i)$$

$$\theta^k \geq 0 \text{ for } k = 1, \ldots, n \quad (5.32j)$$

$$\mu_i^{out} \lambda_{it}^{out,k} + \sum_{(i,j) \in A^{R,out}: z_{ij}^{k,out} = 1} w_{ij(t+\delta_{ij}^{z,out})}^{out,k} \geq \sum_{(i,h) \in A} \gamma_{ht} \quad (5.32k)$$

$$\text{for } k = 1, \ldots, n, i \in T, t \in 1, \ldots, \tau - \delta_i^{max,out}$$

$$\mu_j^{in} \lambda_{jt}^{in,k} + \sum_{(i,j) \in A^{R,in}: z_{ij}^{k,in} = 1} w_{ij(t+\delta_{ij}^{z,in})}^{in,k} \geq \sum_{(h,j) \in A} \gamma_{ht} \text{ for } k = 1, \ldots, n, j \in V, t \in 1, \ldots, \tau - \delta_i^{max,in} \quad (5.32l)$$

$$\lambda_{it}^{out,k} \leq \frac{\sum_{(i,j) \in A^{R,out}: z_{ij}^{k,out} = 1} w_{ij(t+\delta_{ij}^{z,out})}^{out,k}}{\sum_{(i,j) \in A^{R,out}: z_{ij}^{k,out} = 1} \bar{z}_{ij}^{out,k}} \text{ for } k = 1, \ldots, n, i \in T, t = 1, \ldots, \tau \quad (5.32m)$$

$$\lambda_{it}^{in,k} \leq \frac{\sum_{(i,j) \in A^{R,in}: z_{ij}^{k,in} = 1} w_{ij(t+\delta_{ij}^{z,in})}^{in,k}}{\sum_{(i,j) \in A^{R,in}: z_{ij}^{k,in} = 1} \bar{z}_{ij}^{in,k}} \text{ for } k = 1, \ldots, n, i \in V, t = 1, \ldots, \tau \quad (5.32n)$$

$$w_{ij}^{out,k} \geq w_{ij(t-1)}^{out,k} \text{ for } k = 1, \ldots, n, (i,j) \in A^{R,out}, t \in 2 \ldots, \tau \quad (5.32o)$$

$$w_{ijt}^{in,k} \geq w_{ij(t-1)}^{in,k} \text{ for } k = 1, \ldots, n, (i,j) \in A^{R,in}, t \in 2 \ldots, \tau \quad (5.32p)$$

$$w_{\alpha jt}^{out,k} \geq y_i \text{ for } k = 1, \ldots, n, (j,i) \in T^R, t \in \delta_i^y + \delta_{\alpha j}^{z,out}, \ldots, \tau \quad (5.32q)$$

$$w_{\alpha j(t+\delta_{\alpha j}^{z,out})}^{out,k} \geq y_i \text{ for } k = 1, \ldots, n, (j,i) \in B^R \text{ s.t. } z_{\alpha j}^{out,k} = 1, t \in \delta_i^y + 1, \ldots, \tau - \delta_{\alpha j}^{z,out} \quad (5.32r)$$

$$w_{ijt}^{out,k} \geq \bar{z}_{ij}^{out,k} \text{ for } k = 1, \ldots, n, i \in B, j \in V \text{ s.t. } (i,j) \in A^{R,out} \quad (5.32s)$$

$$w_{ijt}^{out,k} \geq \bar{z}_{ij}^{out,k} \text{ for } k = 1, \ldots, n, i \in V, j \in V \text{ s.t. } (i,j) \in A^{R,out}, \exists l \in B, (i,l) \in B^R \quad (5.32t)$$

$$\text{Constraints } (5.10) - (5.12) \quad (5.32u)$$

$$\gamma_{it} = 0 \text{ for } i \in N, t = 1, \ldots, \delta_i^y \quad (5.32v)$$

$$\gamma_{it} = y_i \text{ for } i \in N, t \in (\delta_i^y + 1), \ldots, \tau \quad (5.32w)$$

$$y, \in \{0,1\}^{|N| \setminus \{\alpha,\omega\}} \quad (5.32x)$$

$$\gamma \in \{0,1\}^{|N| \setminus \{\alpha,\omega\} \times \tau} \quad (5.32y)$$

We focus on describing the constraints regarding partial information, since the other constraints are a formulation of the minimum cut problem or are the constraints regarding interdiction. Constraints (5.32k)-(5.32l) allow for restructurings from the k^{th} restructuring plan to be implemented after interdictions that allow for them have occurred. When a node is allowed to restructure more arcs than the number of arcs that node restructured in the k^{th} restructuring plan, constraints (5.32m)-(5.32n) allow for $\lambda = 1$ to ensure the feasibility of constraints (5.32k)-(5.32l). Constraints (5.32o)-(5.32p) enforce that an arc is online in time periods after the time period it was restructured in. Constraints (5.32q) enforce that a back-up trafficker is restructured to when the primary trafficker is interdicted. Likewise, constraints (5.32r) enforce that the victim who was promoted to be the bottom is again promoted when the current bottom is interdicted. Constraints (5.32s)-(5.32t) enforce that any restructured arcs that are independent of interdictions are considered to be restructured.

We note that there are bilinear terms $w^{out,k}_{\alpha jt}\theta^k_{\alpha jt}$ in the objective function constraints, and these terms can be linearized using the McCormick inequalities [84].

After solving (5.32), we identify optimal interdiction decisions $\bar{y}$ for known restructuring plans $(z^1, \zeta^1, \ldots, z^n, \zeta^n)$, as well as a lower bound on the objective value of (5.31), $\bar{\eta}$. We then need to determine the optimal restructuring plan responding to $(\bar{y}, \bar{\gamma})$. To do so, we solve the defender's problem using $(\bar{y}, \bar{\gamma})$ as data.

$$\max_{x,z,\zeta} \sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha it} \tag{5.33a}$$

$$\text{s.t.} \sum_{(h,i) \in A \cup A^{R,out}} x_{hit} = x_{it} \text{ for } i \in N \setminus \alpha, t = 1, \ldots, \tau \tag{5.33b}$$

$$x_{it} = \sum_{(i,h) \in A \cup A^{R,out}} x_{iht} \text{ for } i \in N \setminus \omega, t = 1, \ldots, \tau \tag{5.33c}$$

$$0 \leq x_{ijt} \leq u_{ij} \text{ for } (i,j) \in A, t = 1, \ldots, \tau \tag{5.33d}$$

$$0 \leq x_{ijt} \leq u_{ij}\zeta^{out}_{ij} \text{ for } (i,j) \in A^{R,out} \setminus A^{R,in}, t = 1, \ldots, \tau \tag{5.33e}$$

$$0 \leq x_{ijt} \leq u_{ij}(\zeta^{out}_{ijt} + \zeta^{in}_{ijt}) \text{ for } (i,j) \in A^{R,in}, t = 1, \ldots, \tau \tag{5.33f}$$

$$0 \leq x_{it} \leq u_i(1 - \gamma_{it}) \text{ for } i \in N \setminus \{j \in V : \exists h \in B \text{ with } (h,j) \in B^R\}, t = 1, \ldots, \tau \tag{5.33g}$$

$$0 \leq x_{jt} \leq u_j(1 - \gamma_{jt}) + \tilde{u_j}\zeta^{out}_{\alpha jt} \text{ for } j \in V \text{s.t.} \exists i \in B, (i,j) \in B^R, t = 1, \ldots, \tau \tag{5.33h}$$

$$z, \zeta \in Z(y) \tag{5.33i}$$

Solving (5.33) provides the optimal restructuring plan $\bar{z}$, as well as an upper bound on the objective value of (5.31). Let $U = \min\{U, \sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha it}\}$ be the upper bound on (5.31), let $L = \bar{\eta}$ be the lower bound on (5.31), and let $\epsilon \geq 0$ be the desired error tolerance. If $U - L \leq \epsilon$, then we have identified the desired solution, and the algorithm will terminate. Otherwise, we set $z^{n+1} = \bar{z}$, and repeat the process. We formalize this in Algorithm 10.

5.5.1 Model Simplifications: Upfront Interdiction

We note that, if $\delta^y_i = 0$ for all $i \in N$, meaning that all interdictions are implemented before the network is operated by the defender, we can reduce the number of variables in the constraints. We no longer need γ, as either a node will either be online or offline for the entire time horizon. Similarly, we no longer need to index w^{out}, w^{in} by time, as a restructured arc (i,j) will be in the network in time period $\delta^z_{ij} + 1$. This is reflected by only including the equivalent constraints for (5.32k)-(5.32n) from $\delta^{z,out}_{ij} + 1$ (or $\delta^{z,in}_{ij}$, respectively) to τ. This will only enforce the relationship between whether or not the arc is in the cut (if the arc is

Algorithm 10 C&CG for MP-MFNIP-R

Initialize: lower bound $L = -\infty$, upper bound $U = \infty$, optimal interdiction decisions $(y^*, \gamma^*) = 0$, optimal restructuring decisions $(z^1, \zeta^1) = (z^*, \zeta^*) = 0$, error tolerance $\epsilon \geq 0$. iteration counter $m = 1$.
while $U - L > \epsilon$ **do**
Step 1. Solve (5.32) for optimal interdiction decision (y^n, γ^n) and objective value η. Set $L = \eta$.
if $U - L \leq \epsilon$ **then**
terminate; (y^*, γ^*) and (z^*, ζ^*) are the optimal decisions with objective value U.
end if
Step 2. Input (y^*, γ^*) into (5.33) as data and solve (5.33) for optimal restructuring decisions (z^{n+1}, ζ^{n+1}) and objective value $\sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha i t}$.
if $\sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha i t} < U$ **then**
Let $y^* = y^n$, $\gamma^* = \gamma^n$, $z^* = z^{n+1}$, $\zeta^* = \zeta^{n+1}$, $U = \sum_{t=1}^{\tau} \sum_{i \in N:(\alpha,i) \in A \cup A^{R,out}} x_{\alpha i t}$.
end if
Step 3. Include constraints corresponding to (z^{n+1}, ζ^{n+1}) in (5.32), create variables π^{n+1}, θ^{n+1}, w^{n+1}, set $n = n + 1$, return to Step 1.
end while

restructured in the network) and which side of the cut the nodes are on in the time periods that the arc will appear in.

$$\min_{y,\pi,\theta} \eta \tag{5.34a}$$

$$\text{s.t. } \eta \geq \sum_{t=1}^{\tau} \sum_{i \in N \setminus \{\alpha,\omega\}} u_i \theta_{it}^k + \sum_{(i,j) \in A \cup A^{R,out} \cup A^{R,in}} u_{ij} \theta_{ijt}^k + \sum_{(i,j) \in B^R} \tilde{u_j} z_{\alpha jt}^{out,k} w_{\alpha j}^{out,k} \theta_{\alpha jt}^k \tag{5.34b}$$

$$\text{for } k = 1, \ldots, n$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq 1 \text{ for } k = 1, \ldots, n, (\alpha, j) \in A, t = 1, \ldots, \tau \tag{5.34c}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq 0 \text{ for } (i,j) \in A \text{ s.t. } k = 1, \ldots, n, i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau \tag{5.34d}$$

$$\pi_{it}^{-^k} - \pi_{it}^{+^k} + \theta_{it}^k \geq -y_i \text{ for } k = 1, \ldots, n, i \in N \setminus \{\alpha, \omega\}, t = 1, \ldots, \tau \tag{5.34e}$$

$$-\pi_{it}^{-^k} + \theta_{i\omega t}^k \geq 0 \text{ for } k = 1, \ldots, n, (i, \omega) \in A, t = 1, \ldots, \tau \tag{5.34f}$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq w_{\alpha j}^{out,k} + z_{\alpha jt}^{out,k} - 1 \text{ for } k = 1, \ldots, n, (\alpha, j) \in A^{R,out}, t = 1, \ldots, \tau \tag{5.34g}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq w_{ij}^{out,k} + z_{ijt}^{out,k} - 2 \tag{5.34h}$$

$$\text{for } k = 1, \ldots, n, (i,j) \in A^{R,out} \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq w_{ij}^{in,k} + z_{ijt}^{in,k} - 2 \tag{5.34i}$$

$$\text{for } k = 1, \ldots, n, (i,j) \in A^{R,in} \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau$$

$$\theta^k \geq 0 \text{ for } k = 1, \ldots, n \tag{5.34j}$$

$$\text{Constraints (5.32k)} - \text{(5.32t)} \tag{5.34k}$$

$$\sum_{i \in T} \tilde{r}_i y_i + \sum_{j \in B \cup T} r_i y_i \leq b^a \tag{5.34l}$$

$$\tilde{r}_i \geq r_i^{min} \text{ for } i \in T \tag{5.34m}$$

$$\tilde{r}_i \geq r_i - \sum_{l \in B \cup V} d_l^i y_i \text{ for } i \in T \tag{5.34n}$$

$$y, \in \{0,1\}^{|N| \setminus \{\alpha,\omega\}} \tag{5.34o}$$

Furthermore, we are able to reduce the size of the overall problem due to upfront interdiction. Since the time periods that the restructurings occur in are no longer dependent on the time periods of the interdictions that allowed for those restructurings to occur in, we can divide the time horizon into network "states" based on when multiple sequential time periods do not have any new restructurings. Given a restructuring plan z, let $t_1, \ldots, t_f$ be the *unique* time periods that restructurings in z^k occur in. We can define network state s as $G^s = (V(G), E(G) \cup \{(i,j) \in A^{R,out} : z_{ij}^{k,out} = 1, \delta_{ij}^z \leq t_i\} \cup \{(i,j) \in A^{R,out} : z_{ij}^{k,in} = 1, \delta_{ij}^z \leq t_i\}$. We then need to divide the time horizon $\{1, \ldots, \tau\}$ into S^k intervals in which we are in the different network states. Let $\tau_s^k = t_1 - 1$ for $s = 1$, $\tau_s^k = t_{s+1} - t_s$ for $s = 2, \ldots, t_f$, and $\tau_s^k = \tau - t_f - 1$ for $s = S^k$.

$$\min_{y,\pi,\theta} \eta \tag{5.35a}$$

$$\text{s.t. } \eta \geq \sum_{s=1}^{S^k} \tau_s^k [\sum_{i \in N \setminus \{\alpha,\omega\}} u_i \theta_{is}^k + \sum_{(i,j) \in A \cup A^{R,out} \cup A^{R,in}} u_{ij} \theta_{ijs}^k + \sum_{(i,j) \in B^R} \tilde{u_j} z_{\alpha js}^{out,k} w_{\alpha j}^{out,k} \theta_{\alpha j}^k] \tag{5.35b}$$

$$\text{for } k = 1, \ldots, n$$

$$\pi_{js}^{+^k} + \theta_{\alpha js}^k \geq 1 \text{ for } k = 1, \ldots, n, (\alpha, j) \in A, s = 1, \ldots, S^k \tag{5.35c}$$

$$\pi_{js}^{+^k} - \pi_{is}^{-^k} + \theta_{ijs}^k \geq 0 \text{ for } (i,j) \in A \text{ s.t. } k = 1, \ldots, n, i \neq \alpha, j \neq \omega, s = 1, \ldots, S^k \tag{5.35d}$$

$$\pi_{is}^{-^k} - \pi_{is}^{+^k} + \theta_{is}^k \geq -y_i \text{ for } k = 1, \ldots, n, i \in N \setminus \{\alpha, \omega\}, s = 1, \ldots, S^k \tag{5.35e}$$

$$-\pi_{is}^{-^k} + \theta_{i\omega s}^k \geq 0 \text{ for } k = 1, \ldots, n, (i, \omega) \in A, s = 1, \ldots, S^k \tag{5.35f}$$

$$\pi_{js}^{+^k} + \theta_{\alpha js}^k \geq w_{\alpha j}^{out,k} + z_{\alpha js}^{out,k} - 1 \text{ for } k = 1, \ldots, n, (\alpha, j) \in A^{R,out}, s = 1, \ldots, S^k \tag{5.35g}$$

$$\pi_{js}^{+^k} - \pi_{is}^{-^k} + \theta_{ijs}^k \geq w_{ij}^{out,k} + z_{ijs}^{out,k} - 2 \tag{5.35h}$$

$$\text{for } (i,j) \in k = 1, \ldots, n, A^{R,out} \text{ s.t. } i \neq \alpha, j \neq \omega, s = 1, \ldots, S^k$$

$$\pi_{js}^{+^k} - \pi_{is}^{-^k} + \theta_{ijs}^k \geq w_{ij}^{in,k} + z_{ijs}^{in,k} - 2 \tag{5.35i}$$

$$\text{for } k = 1, \ldots, n, (i,j) \in A^{R,in} \text{ s.t. } i \neq \alpha, j \neq \omega, s = 1, \ldots, S^k$$

$$\theta^k \geq 0 \text{ for } k = 1, \ldots, n \tag{5.35j}$$

$$\text{Constraints (5.32k)} - \text{(5.32t)} \tag{5.35k}$$

$$\sum_{i \in T} \tilde{r}_i y_i + \sum_{j \in B \cup T} r_i y_i \leq b^a \tag{5.35l}$$

$$\tilde{r}_i \geq r_i^{min} \text{ for } i \in T$$

$$\tilde{r}_i \geq r_i - \sum_{l \in B \cup V} d_l^i y_i \text{ for } i \in \tag{5.35m}$$

$$y, \in \{0,1\}^{|N|\setminus\{\alpha,\omega\}} \tag{5.35n}$$

5.6 Augmentations

We seek to improve our C&CG algorithm with modeling-based improvements. The first improvement is to augment previously visited restructuring plans. We do so by including additional arcs as "previously restructured" while still maintaining feasibility. This approach helps to better take advantage of the way partial information about previously visited restructured plans provide bounds on the current interdiction decisions. Note that, as long as a restructuring plan satisfies the constraints that are independent of y, then the w variables will ensure that the constraints that are dependent on y are satisfied, producing a feasible restructuring plan. As such, any previously visited restructuring plan can be augmented to include additional arcs *as long as* the constraints independent of y are satisfied.

We provide a simple, yet effective augmentation. Every previously visited restructuring plan (including the initial iteration: the empty restructuring plan) is augmented to include every backup trafficker. That is, for every $k = 1, \ldots, n$, $(j,i) \in T^R$, $\bar{z}_{\alpha j}^{out,k} = 1$. Note that the only constraints on whether or not a backup trafficker can be restructured are the constraints indicating whether or not the primary trafficker has been interdicted. Thus, this change will not impact the feasibility of any constraints independent of y. Since, for $(i,j) \in T^R$, $w_{\alpha jt}^{out,k} = 0$ for all t if $y_i = 0$, and $w_{\alpha jt}^{out,k} = 1$ for $t \geq \delta_i^y + \delta_{\alpha j}^{z,out} + 1$ if $y_i = 1$, the resulting restructuring sub plan will be feasible. This augmentation will allow the master problem to better project what the impact on flow will be if a trafficker is interdicted, which will in turn produce better lower bounds when traffickers are interdicted.

The next improvement we propose is to project how new victims may be recruited. For each previously considered restructuring plan, we can identify if a prospective victim j has been recruited or not by computing $\sum_{i \in T:(i,j) \in A^{R,out}} \bar{z}_{ij}^{out,k}$. If $\sum_{i \in T:(i,j) \in A^{R,out}} \bar{z}_{ij}^{out,k} = 1$, then prospective victim j has been recruited in restructuring plan m. Otherwise, $\sum_{i \in T:(i,j) \in A^{R,out}} \bar{z}_{ij}^{out,k} = 0$, meaning that prospective victim j has not been recruited. We can thus identify the set of prospective victims that have not yet been recruited as $P^k = \{j \in V^R : \sum_{i \in T:(i,j) \in A^{R,out}} \bar{z}_{ij}^{out,k} = 0\}$. Additionally, we can identify the *latest* a victim can be recruited as $\bar{t} = \max_{i \in B \cup V} \delta_i^y + \max_{(i,j) \in A^{R,out}: j \in P^k} \delta_{ij}^{z,out}$. Likewise, we can identify which traffickers still have the ability to act (recall that constraints (5.22) restrict the number of acts each trafficker can take, which is independent of the interdiction decisions). Let

$a_i^k = \sum_{j \in N:(i,j) \in A_R^{out}} \bar{z}_{ij}^{out} + \sum_{\substack{l \in N, h \in N:(i,l) \in A \\ (l,h) \in A_R^{out}}} \bar{z}_{lh}^{out} + \sum_{\substack{l \in N:(i,l) \in A \\ (\alpha,l) \in A_R^{out}}} \bar{z}_{\alpha l}^{out}$ be the number of actions taken by trafficker i in restructuring plan m. Note that if $a_i^k = c_i^{out}$, then trafficker i has cannot taken any more actions in restructuring plan k. However, if $a_i^k < c_i^{out}$, then trafficker i may still be able to recruit new victims. Let $C^k = \{i \in T : a_i^k < c_i^{out}\}$ be the set of traffickers that can still recruit in restructuring plan k. From these two sets, we can identify opportunities for recruitment. Let $A^{k,rec} = \{(i,j) \in A^{R,out} : i \in C^k, j \in P^k\}$ be the set of restructurable arcs (i,j) where a trafficker i that can still act recruits prospective victim j. For $k = 1, \ldots, n$, $(i,j) \in A^{k,rec}$, let ψ_{ijt}^k be the indicator variable as to whether or not (i,j) may be restructured in addition to restructuring plan k in time period t. We want to define constraints that enforce that, if more interdictions have occurred that would allow for more restructurings to occur, arcs in $A^{k,rec}$ can additionally be restructured to augment that restructuring plan. In order to do so, we need to define two additional variables to ensure feasibility. Let ν_{ki} be the indicator of whether or not trafficker i has performed c_i^{out} actions (including projected recruitment) in plan k, and let ξ_{ki} be the indicator of whether or not trafficker i has restructured all of the arcs they can in $A^{k,rec}$. We can define the following constraints:

$$M(\nu_{ki} + \xi_{ki}) + \sum_{j \in P^k:(i,j) \in A^{k,rec}} \psi_{ij\bar{t}}^k \geq \sum_{(i,j) \in A} y_j - \sum_{(i,j) \in A^{R,out}} w_{ij\bar{t}}^{out,k} \tag{5.36}$$
$$\text{for } k = 0, \ldots, n, i \in C^k$$

$$\sum_{j \in P^k:(i,j) \in A^{k,rec}} \psi_{ij\bar{t}} \leq c_i^{out} - a_i^k \tag{5.37}$$
$$\text{for } k = 0, \ldots, n, i \in C^k$$

$$\nu_{ki} \leq \frac{1}{c_i^{out}} \left(\sum_{\substack{j \in N:(i,j) \in A_R^{out} \\ \bar{z}_{ij}^{k,out} = 1}} \bar{w}_{ij\tau}^{out,k} + \sum_{\substack{l \in N, h \in N:(i,l) \in A \\ (l,h) \in A_R^{out} \\ \bar{z}_{lh}^{out,k} = 1}} \bar{w}_{lh\tau}^{out,k} + \sum_{\substack{l \in N:(i,l) \in A \\ (\alpha,l) \in A_R^{out} \\ \bar{z}_{\alpha l}^{k,out} = 1}} \bar{w}_{\alpha l\tau}^{out,k} + \sum_{j \in P^k:(i,j) \in A^{k,rec}} \psi_{ij\bar{t}}^k \right) \tag{5.38}$$
$$\text{for } k = 1, \ldots, n, i \in C^k$$

$$\xi_{ki} \leq \frac{\sum_{j \in P^k:(i,j) \in A^{k,rec}} \psi_{ij\bar{t}}^k}{|j \in P^k : (i,j) \in A^{k,rec}|} \tag{5.39}$$
$$\text{for } k = 1, \ldots, n, i \in C^k$$

$$\psi_{ijt}^k \geq \psi_{ij(t-1)}^k \tag{5.40}$$
$$\text{for } (i,j) \in A^{k,rec}, t \in (\bar{t}+1), \ldots, \tau$$

Constraints (5.36) enforce that, for each previously considered restructuring plan, ad-

ditional recruitment occurs when more interdictions occur than restructurings that were enabled by interdictions. Constraints (5.37) enforce that every trafficker takes at most c_i^{out} action. Constraints (5.38) enforce that c_i^{out} actions are taken before allowing $\nu_{ki} = 1$, which ensures the feasibility of (5.36) when more than c_i^{out} interdictions disrupt nodes incident to trafficker i. Likewise, constraints (5.39) enforce that all possible potential recruitment occurs before allowing $\xi_{ki} = 1$, which ensures the feasibility of (5.36) when all $\psi^k_{ij\bar{t}} = 1$ for all $j \in P^k : (i,j) \in A^{k,rec}$, ie. all potential recruitment opportunities are taken. Constraints (5.40) ensure that an arc stays online once it comes online.

We note that the introduction of these ψ variables may cause constraints (5.23) to be infeasible. However, in our model, $u_j = 1$ for all $j \in V^R$. This allows us to show that, even if the constraints are violated, it will not impact the minimum cut in the network. First, we define what it means for a node to be in the minimum cut.

Definition 3. A node $i \in N$ is in the minimum cut at time t if $\theta_{it} = 1$.

Theorem 7. For every restructuring plan k, every prospective victim $v \in P^k$ and time period $t \geq \bar{t}$, at most one trafficker node i that is not in the minimum cut will be brought into the minimum cut when $\psi^k_{ivt} = 1$, regardless of the number of trafficker nodes where $\psi^k_{ivt} = 1$.

Proof. Suppose that, with restructuring plan k in time period $\bar{t} - 1$, there are two non-interdicted trafficker nodes where $(i,v) \in A^{k,rec}$ and $(j,v) \in A^{k,rec}$, and $\psi^k_{ivt} = \psi^k_{jvt} = 1$. Let $f^k_{\bar{t}-1}$ be the maximum flow in through the network with restructuring plan k at time period $\bar{t} - 1$ as determined by the Ford-Fulkerson algorithm, and let $(\bar{\pi}^k_{\bar{t}-1}, \bar{\theta}^k_{\bar{t}-1})$ be the minimum cut in time period $\bar{t} - 1$ as determined by the residual network D_f, and suppose that $\theta^k_{i(\bar{t}-1)} = \theta^k_{j(\bar{t}-1)} = 0$. Thus, there are at least two $\alpha - \omega$ augmenting paths in time period $\bar{t}$, (α, i, v, ω) and (α, j, v, ω). Only one augmenting path through v can be chosen, since $u_v = 1$ will be amount of flow sent along the chosen path due to node and arc capacities being integral. Thus, if j is not in the chosen path, then whether or not j can be reached by α will not be impacted by the inclusion of arc (j,v) after flow is sent along the augmenting path using i. Thus, at most one trafficker node will be included in the minimum cut after including arcs (i,v) and (j,v). □

Since at most one $i \in T$ will be included in the minimum cut, regardless of the number of arcs to recruitable victims are "included" via ψ variables, we can construct a feasible

restructuring plan. If there were to be a change in the minimum cut after "including" an arc (i, j) via $\psi^k_{ijt} = 1$, then (i, j) would be the arc that would be restructured. If no such change occurs, then any one arc can be chosen and the minimum cut will only increase by 1, regardless of which arc is chosen. Removing the extra arcs introduced by ψ will not impact the feasibility of any of the restructuring plan. We also note that, the only arcs that are added after the projected recruitment arcs will be restructuring to a back-up trafficker, promoting a victim to be the new bottom, and assigning the newly promoted bottom more victims. For a trafficker that was not interdicted, these arcs do not increase the total number of victims a trafficker can reach. Thus, if a victim is recruited into a given organization, no arcs restructured after their recruitment will cause them to be recruited into a different organization, and thus a feasible restructuring plan can be constructed. This augmentation can be implemented in both the standard model and simplified upfront interdiction models. To implement the prospective recruitment constraints in the network state model, we would need to introduce an extra state starting at $\bar{t}$ if no such state where to exist.

5.7 Computational Results

We implement the models derived for Algorithm 10 in AMPL with Gurobi 9.5 as the solver. Experiments were conducted on a laptop with an Intel® Core™ i5-8250 CPU @ 1.6 GHz - 1.8 GHz and 16 GB RAM running Windows 10. Each instance is limited to a run time of 2 hours.

We test out model using $\tau = 7$ on the networks used in Chapter 4. As in Chapter 4, we choose the cost to interdict a victim to be 2, the cost to interdict a bottom is 4, and the cost to interdict a trafficker is 8. Interdicting a victim reduces the cost to interdict their trafficker by 1, while interdicting a bottom reduces the cost by 3, to a minimum of 4. In the models with delayed interdiction, interdicting a victim takes 1 time period, interdicting a bottom takes 2 time periods, and interdicting a trafficker takes 3 time periods. In models with a second attacker, the second attacker has a fixed budget of 10, where interdicting a victim already in the trafficking network costs 3, and preventing a victim from being recruited costs 1.

Each trafficker may restructure up to 4 arcs in their operation, while each victim may restructure 1 arc if their trafficker is interdicted. Restructuring to a victim currently in the trafficking network takes 1 time period, and recruiting a new victim takes 2 time periods.

A trafficker taking a victim from their bottom or assigning a victim to their bottom also takes 1 time period. Trafficking operations with at least 4 victims (including a bottom, if the operation has one) will have a back-up trafficker, and it takes 2 time periods to restructure to them. For operations with a bottom, the set of victims that can be promoted to be the new bottom is determined by randomly selecting half of the victims the trafficker is adjacent to (rounding up), and promoting a victim to be a bottom takes 3 time periods. We set the number of recruitable victims to be 40% of the number of victims in the network.

5.7.1 Comparison of Solution Methods

We first compare the solve times of Algorithm 10 with and without the augmentations for models with a single attacker. Table 5.1 compares the solve times for delayed interdictions. Table 5.2 compares solves times for the base formulation of upfront interdiction, and Table 5.3 compares the solve times for the network state formulation of upfront interdiction. In each table, we report the solve time and number of restructuring plans visited. Unsolved instances are marked with an $*$.

From these tables, we can see that, in the models with a single attacker, the modeling-based augmentations allow the C&CG algorithm to almost always solve the problem significantly quicker for all formulations. Out of 135 instances tested, only 6 instances are solved faster without the augmentations. In all but two of these instances, the solve time for both methods is under 10 seconds, indicating that those instances can already be easily solved. For all the instances for delayed interdictions, the method with augmentations always outperform the base method. The improvement in solve time by including the augmentations is quite significant. In 114 out of 135 instances, the solve time of the method with augmentations is at most 50% that of the base method, and in 39 out of 135 instances, the solve time is at most 10% that of the base method. Additionally, the base method is unable to solve 19 instances, whereas the method with augmentations solves every instance. In comparing Tables 5.2 and 5.3, we can additionally see that the network state formulation for upfront interdiction drastically outperforms the base formulation. Every instance that both formulations solve is solved quicker with the network state formulation, with 68 out of 90 instances being solved in at most 50% of the solve time of the base formulation. Additionally, the network state formulation solves two more instances than the base formulation.

We now present our results for the models with two cooperating attackers. Table 5.4

compares solve times with delayed interdictions, Table 5.5 compares solves times for the base formulation of upfront interdiction, and Table 5.6 compares the solve times for the network state formulation of upfront interdiction.

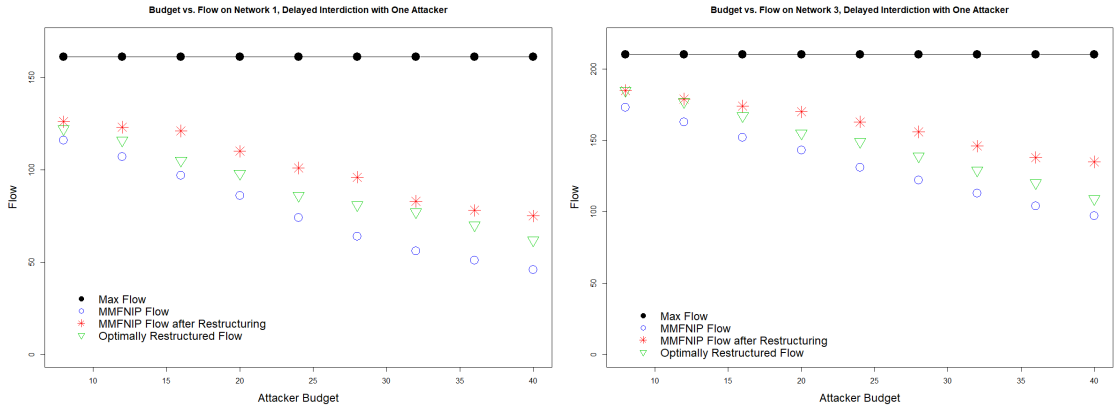
Here, we can see that, while the method with augmentations is less effective with two attackers than with one, it often solves the problem faster than the method without augmentations. We expect that the augmentations would be less effective in these instances, since the second attacker can prevent the recruitment of new participants, which would limit the benefits of the augmentation that projects the recruitment of new participants. Now, 28 of the instances are solved faster without including augmentations. In instances with two attackers, the base method is unable to solve 8 instances, and the method with augmentations is unable to solve 4 instances. We note that, every instance that is solved by the base method is also solved by the method with augmentations. In comparing Tables 5.5 and 5.6, we can again see the significant benefits of the network state formulation for upfront interdiction. The base formulation only outperforms the network state formulation in 3 instances, all of which are solved in under 60 seconds. For the network state formulation, 54 of the 90 instances are solved in at most 50% of the solve time of the base formulation. Additionally, the network state formulation is able to solve every instance, where the base formulation is unable to solve 5 instances

We now compare the quality of the bounds determined in unsolved instances. For instances with delayed interdiction for one attacker and two attackers, we report the relative optimality gap, the difference between the upper and lower bound divided by the upper bound, in Tables 5.7 and 5.8. For instances with upfront interdiction for one attacker and two attackers, we report the relative optimality gap in Tables 5.9 and 5.10. These results show that, with one exception, the quality of the bounds identified using the method with augmentations are higher quality than the base method, and that the bounds identified by the network-state model for upfront interdiction are higher quality than the base model.

5.7.2 Application-Based Analysis

We now compare the policy recommendations of a multi-period MFNIP model without our restructuring against those recommended by MP-MFNIP-R. For unsolved instances, we present the solution associated with the upper bound from the model and method that resulted in the smallest upper bound. This is because this solution is a bilevel feasible solution,

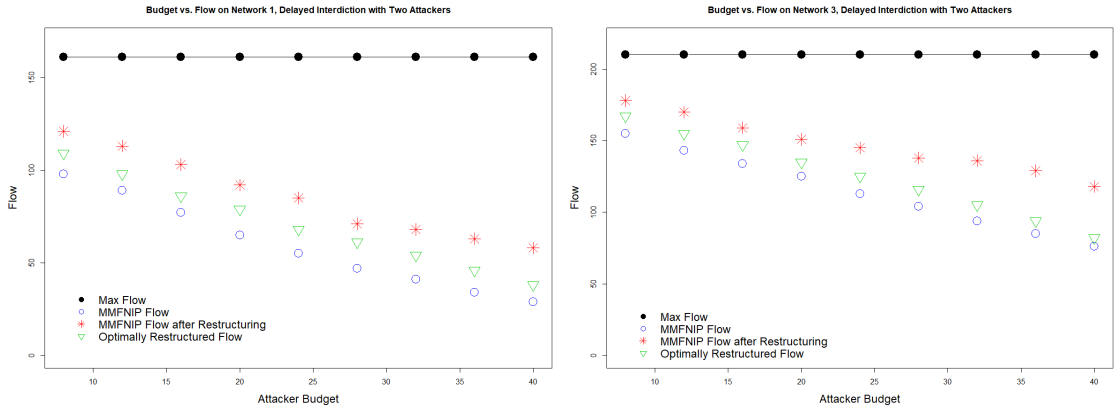
and thus is what we want to minimize. We present the results of two sample networks, network 1 and 3, as the other three follow similar follow similar trends to one of these two; those results are included in Appendix G. We first present the results on flow through the network. Figures 5.1 and 5.2 present the flows with delayed interdictions with one and two attackers, respectively. Figures 5.3 and 5.4 present the flows with delayed interdictions with one and two attackers, respectively. In these figures, the black dotted line represents the total flow through the un-interdicted network, which is the total number of victims (bottoms included) multiplied by the number of time periods. The blue circles are the optimal flow as determined by multi-period MFNIP, and the red stars are the optimal flow after restructuring responding to multi-period MFNIP's recommended interdictions. The green triangles are the optimal flow as determined by MP-MFNIP-R.



(a) **(b)**

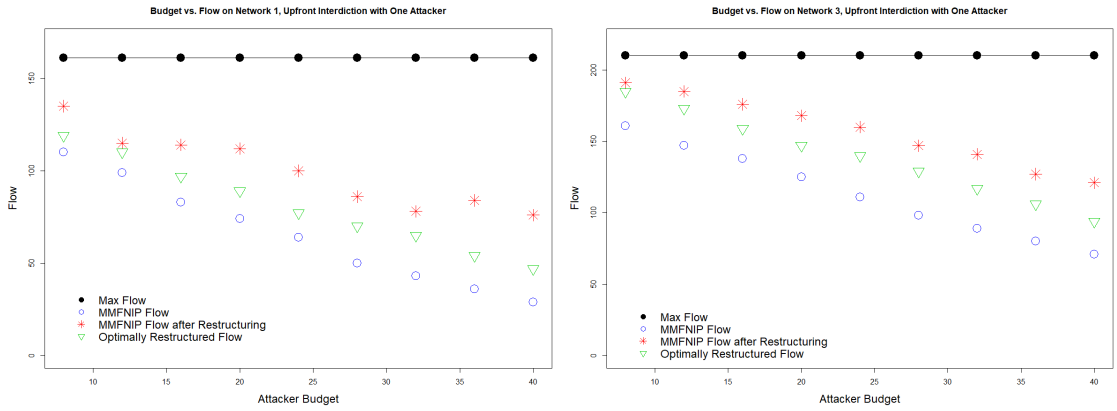
Figure 5.1: Interdicted and restructured flows with delayed interdiction and one attacker over varying attacker budgets one on network 1 (a) and network 3 (b)

Across all instances, the defender is able to regain a significant amount of flow after restructuring in response to the interdictions recommended by multi-period MFNIP. However, the interdictions recommended by MP-MFNIP-R are able to account for these restructurings, resulting in a lower optimal flow. However, in instances with only one attacker, these flows are closer to the restructured flows than the flows projected by multi-period MFNIP. This is particularly noticeable in network 1 at higher budget levels, and network 3 at lower budget levels. However, with the inclusion of a second attacker that can prevent the recruitment of new participants, the amount of flow restructured in MP-MFNIP-R is significantly limited, especially in the case of network 3. By preventing recruitment, more victims can be



(a) (b)

Figure 5.2: Interdicted and restructured flows with delayed interdiction and two attackers over varying attacker budgets on network 1 (a) and network 3 (b)



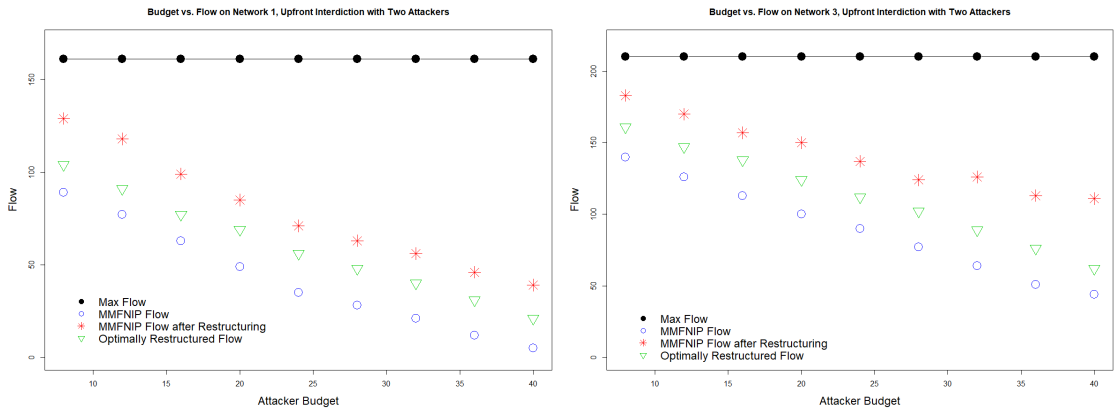
(a) (b)

Figure 5.3: Interdicted and restructured flows with upfront interdiction and one attacker over varying attacker budgets on network 1 (a) and network 3 (b)

interdicted, while also preventing them from being replaced, resulting in effectively reducing the total flow.

In comparing the flow differences between delayed interdiction and upfront interdiction, the difference in flows between the two models starts small, and decreases in flow from upfront interdictions grows as the attacker budget increases. This aligns with our expectations, since at larger budget levels, the attacker with upfront interdictions can disrupt more of the network sooner. Even though this allows for restructurings to be implemented in earlier time periods, the original network is able to operate better than the restructured network.

We next present results on how many nodes of each type were recommended to be



(a) **(b)**

Figure 5.4: Interdicted and restructured flows with upfront interdiction and two attackers over varying attacker budgets on network 1 (a) and network 3 (b)

interdicted. Tables 5.11 and 5.13 present the recommended interdictions for network 1 with delayed interdictions and one and two attackers, respectively. Tables 5.12 and 5.14 likewise present these results on network 3. Tables 5.15 - 5.18 present these results with upfront interdictions. Columns with "MMFNIP" report the results from multi-period MFNIP, and columns with "MP-MFNIP-R" report the results from MMNFIP-R. Columns with "2nd" report the decisions of the second attacker.

In the case of delayed interdiction with one attacker, multi-period MFNIP prioritizes interdicting bottoms in both networks, while focusing more on interdicting victims in network 1, and more on interdicting traffickers in network 3. Since network 3 has more victims, it would be reasonable that disrupting traffickers would reduce the flow more, even accounting for the time delay. However, in both networks, MP-MFNIP-R consistently allocates a significant portion of the attacker budget to disrupting victims. This is likely due to the inclusion of backup traffickers that the multi-period MFNIP is unable to account for. These trends continue for the case with two attackers. At lower budget levels, the second attacker spends more budget on interdicting current victims, and shifts their focus to preventing the recruitment of new victims at higher budget levels. In both networks, the number of prospective victims that the second attacker prevents their recruitment is double that of the number of current victims they interdict when the primary attacker has a budget of at least 24. As the attacker budget increases, the first attacker is able to disrupt more victims, allowing for more opportunities for recruitment, making the prevention of recruitment a higher priority.

This accounts for the decrease in restructured flow seen in Figures 5.2 and 5.4.

In the case of upfront interdiction with one attacker, the recommended interdictions from MP-MFNIP-R in network 1 drastically shift towards interdicting traffickers and bottoms, while the recommended interdictions from MP-MFNIP-R in network 3 are almost completely consistent with the recommendations from the delayed interdiction model. We additionally have that, when there are two attackers, the first attacker's interdictions increasingly focus on disrupting victims currently in the network, while the second attacker again focuses on the prevention of recruitment. We again observe that, in both networks, the number of prospective victims that the second attacker prevents their recruitment is double that of the number of current victims they interdict when the primary attacker has a budget of at least 24. This suggests that disrupting the ability of traffickers to recruit new victims will be key in effectively disrupting their operations.

5.8 Conclusion

We introduced the multi-period max flow network interdiction problem with restructuring (MP-MFNIP-R), where flow is sent from the source node to sink node in each time period, and interdictions and restructurings are decided upon upfront and implemented throughout the time horizon. We motivated this problem with applications in disrupting domestic sex trafficking networks. We modeled this problem as a bilevel mixed integer-linear program (BMILP), and derived a column-and-constraint generation (C&CG) algorithm to solve this problem. Modeling-specific augmentations were incorporated in the C&CG algorithm to significantly improve the solve time. For models where the interdictions are also implemented upfront, we proposed equivalent models that are able to significantly reduce the size of the problem. We additionally proposed additional models that include a second attacker that has the ability to interdict victims currently in the network, as well as to prevent the recruitment of new victims, modeling the ability of social services to disrupt a sex trafficking network.

We tested our model on synthetic domestic sex trafficking networks with 5 traffickers. We note that the benefits of all interdictions occurring upfront was more noticeable at higher attacker budget levels. Additionally, our results show that, with the inclusion of a second attacker, the ability to prevent recruitment is key to effectively disrupting the operations of these networks. By preventing recruitment, the resources spent on disrupting victims means

that these victims cannot be replaced by traffickers. This suggested that future research, both qualitative and quantitative, should focus on exploring how to disrupt recruitment in sex trafficking networks.

Table 5.1: Comparison of run times (seconds) and number of plans visited of algorithm 10 for delayed interdictions with and without augmentations, one attacker

Budget	Data1, Base	Data1, Aug	Data2, Base	Data2, Aug	Data3, Base	Data3, Aug	Data4, Base	Data4, Aug	Data5, Base	Data5, Aug
8	7.328 (6)	**3.313 (4)**	96.312 (14)	**41.484 (8)**	146.703 (12)	**48.812 (8)**	430.016 (17)	**36.453 (4)**	0.812 (3)	**0.453 (2)**
12	318.421 (19)	**43.656 (8)**	764.110 (17)	**61.922 (6)**	7200* (32)	**4047.500 (18)**	853.891 (15)	**224.391 (6)**	0.891 (3)	**0.484 (2)**
16	18.375 (6)	**16.187 (5)**	1871.375 (19)	**122.703 (6)**	7200* (20)	**1783.781 (9)**	373.859 (10)	**45.937 (3)**	39.031 (11)	**27.719 (10)**
20	119.782 (11)	**45.516 (6)**	2663.765 (18)	**418.422 (8)**	2665.813 (12)	**129.266 (4)**	316.156 (9)	**83.750 (4)**	12.750 (7)	**5.656 (4)**
24	20.157 (6)	**6.860 (4)**	2166.250 (15)	**916.703 (8)**	7200* (15)	**4757.234 (9)**	3631.657 (15)	**788.032 (8)**	11.656 (6)	**3.344 (3)**
28	283.906 (12)	**33.015 (5)**	7200* (18)	**6889.297 (14)**	7200* (14)	**2707.797 (9)**	4164.390 (15)	**677.562 (8)**	232.110 (16)	**57.266 (8)**
32	3101.609 (28)	**699.203 (12)**	7200* (14)	**336.813 (5)**	7200* (14)	**5753.937 (10)**	3811.922 (14)	**643.750 (8)**	719.649 (18)	**85.296 (8)**
36	616.703 (17)	**441.172 (9)**	7200* (16)	**3680.265 (12)**	7200* (12)	**399.141 (5)**	241.766 (7)	**54.219 (4)**	169.546 (11)	**23.579 (6)**
40	237.422 (17)	**31.594 (6)**	6058.062 (18)	**1031.157 (10)**	985.781 (8)	**74.172 (4)**	1101.593 (10)	**420.906 (6)**	186.516 (13)	**20.937 (5)**

Table 5.2: Comparison of run times (seconds) and number of plans visited of algorithm 10 for the base formulation with upfront interdictions with and without augmentations, one attacker

Budget	Data1, Base	Data1, Aug	Data2, Base	Data2, Aug	Data3, Base	Data3, Aug	Data4, Base	Data4, Aug	Data5, Base	Data5, Aug
8	2.719 (5)	**1.312 (3)**	19.812 (9)	**9.157 (4)**	390.796 (25)	**119.454 (9)**	1803.500 (27)	**495.203 (6)**	0.781 (3)	**0.515 (2)**
12	**4.687 (5)**	6.875 (5)	774.750 (22)	**47.500 (5)**	4644.547 (31)	**1191.187 (14)**	**492.172 (16)**	526.922 (6)	1.735 (4)	**0.532 (2)**
16	2.750 (3)	**2.062 (3)**	276.156 (14)	**6.453 (3)**	5407.688 (27)	**191.313 (7)**	318.343 (13)	**128.219 (5)**	25.344 (12)	**5.187 (5)**
20	76.485 (12)	**5.907 (4)**	2759.157 (22)	**93.922 (7)**	1967.437 (19)	**575.296 (6)**	239.688 (11)	**92.391 (4)**	40.015 (13)	**3.344 (3)**
24	28.140 (7)	**6.625 (3)**	3071.187 (21)	**44.328 (4)**	7200* (22)	**4597.032 (10)**	3712.719 (20)	**1016.203 (8)**	11.781 (6)	**1.797 (2)**
28	166.532 (13)	**37.218 (6)**	6786.156 (27)	**874.359 (8)**	7200* (19)	**3297.032 (13)**	2710.734 (17)	**261.094 (5)**	125.844 (14)	**14.093 (5)**
32	1823.703 (28)	**530.125 (14)**	2172.750 (16)	**208.234 (6)**	7200* (19)	**2789.437 (9)**	3801.25 (18)	**254.750 (5)**	1135.031 (25)	**200.625 (14)**
36	284.125 (14)	**98.782 (7)**	3969.797 (18)	**1945.579 (9)**	7200* (17)	**513.078 (5)**	1825.578 (13)	**86.031 (4)**	408.438 (17)	**101.500 (9)**
40	345.781 (16)	**22.750 (7)**	7200* (19)	**3442.734 (12)**	3256.641 (13)	**507.516 (5)**	7200* (17)	**917.297 (8)**	385.359 (18)	**61.985 (8)**

Table 5.3: Comparison of run times (seconds) and number of plans visited of algorithm 10 for the network state formulation with upfront interdictions with and without augmentations, one attacker

Budget	Data1, Base	Data1, Aug	Data2, Base	Data2, Aug	Data3, Base	Data3, Aug	Data4, Base	Data4, Aug	Data5, Base	Data5, Aug
8	2.094 (6)	**0.891 (3)**	8.953 (10)	**2.250 (4)**	71.203 (19)	**29.531 (8)**	765.563 (24)	**65.765 (5)**	**0.266 (2)**	0.312 (2)
12	**2.203 (5)**	2.344 (4)	169.360 (20)	**16.891 (4)**	868.422 (28)	**230.922 (12)**	**166.172 (15)**	228.344 (8)	0.875 (4)	**0.360 (2)**
16	**0.828 (3)**	1.140 (3)	107.765 (16)	**3.031 (3)**	1647.062 (25)	**122.203 (7)**	102.234 (12)	**40.438 (5)**	5.969 (10)	**1.922 (4)**
20	35.562 (12)	**2.422 (4)**	1021.813 (22)	**22.641 (6)**	691.657 (19)	**116.187 (6)**	81.281 (11)	**30.562 (4)**	13.125 (10)	**1.343 (3)**
24	6.422 (6)	**3.438 (4)**	742.937 (19)	**33.484 (5)**	7200* (30)	**727.485 (10)**	1443.250 (20)	**166.578 (7)**	8.375 (7)	**0.610 (2)**
28	40.844 (11)	**22.125 (7)**	2382.922 (25)	**114.781 (6)**	7200* (26)	**420.937 (9)**	1007.984 (18)	**57.828 (5)**	100.375 (17)	**6.375 (5)**
32	279.938 (22)	**148.234 (12)**	1057.359 (18)	**55.313 (5)**	7200* (24)	**556.641 (9)**	947.078 (18)	**85.594 (5)**	377.656 (23)	**31.703 (8)**
36	80.093 (14)	**13.438 (5)**	1031.610 (16)	**345.234 (9)**	5662.719 (22)	**115.781 (5)**	378.156 (13)	**34.031 (5)**	285.000 (21)	**21.515 (6)**
40	118.047 (17)	**14.578 (7)**	7200* (32)	**452.969 (11)**	2914.656 (17)	**96.766 (5)**	2294.422 (19)	**340.813 (9)**	157.781 (19)	**31.672 (8)**

Table 5.4: Comparison of run times (seconds) and number of plans visited of algorithm 10 for delayed interdictions with and without augmentations, two attackers

Budget	Data1, Base	Data1, Aug	Data2, Base	Data2, Aug	Data3, Base	Data3, Aug	Data4, Base	Data4, Aug	Data5, Base	Data5, Aug
8	203.547 (12)	**110.922 (7)**	224.718 (13)	**84.875 (6)**	566.860 (11)	**524.313 (7)**	64.922 (6)	**28.281 (4)**	**30.219 (9)**	62.968 (8)
12	147.344 (10)	**86.906 (6)**	1331.750 (16)	**1044.469 (10)**	**527.109 (10)**	567.937 (7)	124.734 (6)	**114.157 (5)**	46.922 (10)	**16.485 (5)**
16	**9.047 (5)**	9.532 (4)	361.203 (8)	**218.828 (6)**	**923.531 (8)**	3700.844 (10)	**46.657 (5)**	164.375 (6)	79.687 (11)	**8.687 (4)**
20	95.140 (9)	**16.312 (4)**	**809.766 (13)**	2674.500 (13)	1369.375 (8)	**349.516 (6)**	2296.250 (18)	**947.781 (8)**	19.562 (7)	**7.328 (3)**
24	24.125 (7)	**17.141 (5)**	7200* (18)	7200* (13)	3920.375 (14)	**504.828 (7)**	7200* (14)	7200 (11)*	**37.766 (8)**	50.032 (8)
28	**16.813 (6)**	25.390 (6)	7200* (26)	**5060.297 (11)**	**1148.625 (9)**	1748.016 (9)	595.468 (8)	**525.344 (7)**	92.422 (8)	**69.859 (6)**
32	66.515 (11)	**52.094 (8)**	**1509.125 (18)**	2249.719 (11)	3430.563 (15)	**342.312 (6)**	3707.141 (14)	**3673.765 (12)**	14.235 (8)	**2.375 (2)**
36	**24.344 (7)**	30.531 (7)	**116.140 (6)**	158.922 (7)	1557.259 (12)	**148.578 (5)**	**407.844 (9)**	749.954 (9)	3.937 (4)	**2.250 (3)**
40	**2.500 (3)**	5.782 (4)	**32.094 (6)**	571.031 (12)	**32.469 (4)**	34.719 (4)	7200* (21)	7200* (17)	**2.500 (3)**	2.562 (3)

Table 5.5: Comparison of run times (seconds) and number of plans visited of algorithm 10 for the base formulation with upfront interdictions with and without augmentations, two attackers

Budget	Data1, Base	Data1, Aug	Data2, Base	Data2, Aug	Data3, Base	Data3, Aug	Data4, Base	Data4, Aug	Data5, Base	Data5, Aug
8	77.078 (9)	**39.297 (6)**	92.656 (10)	**50.157 (5)**	**383.265 (14)**	495.062 (7)	17.812 (6)	**10.563 (3)**	**11.937 (7)**	52.343 (7)
12	64.329 (9)	**34.172 (6)**	1783.781 (18)	**173.828 (6)**	263.438 (9)	**202.204 (6)**	492.625 (9)	**176.500 (4)**	7.125 (4)	**2.860 (2)**
16	27.484 (6)	**7.734 (4)**	790.297 (10)	**352.578 (8)**	7200* (15)	**1095.687 (6)**	275.500 (8)	**50.250 (4)**	42.891 (10)	**6.672 (3)**
20	153.922 (11)	**37.281 (5)**	4496.516 (23)	**390.344 (8)**	1889.781 (12)	**458.078 (6)**	5703.500 (18)	**3395.328 (11)**	**8.047 (4)**	10.281 (4)
24	37.953 (8)	**12.672 (4)**	7200* (21)	7200* (13)	3934.318 (17)	**2142.922 (9)**	6817.485 (21)	**1811.766 (10)**	59.594 (6)	**6.375 (3)**
28	86.469 (9)	**48.203 (6)**	7200* (30)	**6942.609 (20)**	**3491.125 (13)**	3576.328 (8)	2413.125 (14)	**1128.203 (8)**	154.578 (17)	**42.953 (6)**
32	203.562 (14)	**14.313 (4)**	2200.047 (18)	**1454.110 (11)**	2365.125 (13)	**897.406 (8)**	2729.718 (14)	**486.703 (6)**	9.531 (4)	**5.563 (3)**
36	238.141 (15)	**100.015 (10)**	818.328 (15)	**351.484 (8)**	890.188 (10)	**284.032 (6)**	826.594 (11)	**181.890 (5)**	11.703 (5)	**4.328 (4)**
40	115.562 (20)	**86.985 (16)**	**507.312 (13)**	1362.281 (12)	377.203 (9)	**64.078 (4)**	7200* (24)	**2704.344 (11)**	**10.313 (4)**	10.640 (6)

Table 5.6: Comparison of run times (seconds) and number of plans visited of algorithm 10 for the network state formulation with upfront interdictions with and without augmentations, two attackers

Budget	Data1, Base	Data1, Aug	Data2, Base	Data2, Aug	Data3, Base	Data3, Aug	Data4, Base	Data4, Aug	Data5, Base	Data5, Aug
8	51.515 (10)	**9.781 (6)**	101.985 (11)	**16.375 (5)**	185.360 (12)	**119.578 (6)**	**7.8282 (5)**	13.531 (4)	3.750 (7)	**3.438 (5)**
12	39.922 (10)	**22.235 (7)**	628.718 (18)	**61.641 (5)**	115.203 (10)	**44.016 (4)**	250.797 (11)	**67.615 (5)**	10.110 (8)	**2.765 (3)**
16	**4.297 (4)**	9.250 (5)	242.766 (12)	**45.406 (5)**	1466.735 (16)	**570.734 (8)**	110.766 (8)	**13.219 (3)**	52.422 (11)	**4.735 (3)**
20	55.859 (11)	**7.187 (4)**	576.250 (15)	**203.750 (10)**	884.078 (12)	**240.141 (6)**	2753.535 (21)	**1138.469 (13)**	5.843 (5)	**4.828 (4)**
24	19.766 (9)	**6.250 (4)**	2452.266 (22)	**2021.250 (15)**	**624.765 (12)**	662.359 (10)	1893.531 (19)	**487.640 (9)**	57.782 (9)	**5.140 (4)**
28	52.516 (16)	**5.250 (4)**	3102.656 (31)	**1410.938 (16)**	1220.032 (15)	**817.469 (10)**	454.047 (12)	**169.297 (6)**	90.125 (16)	**22.141 (7)**
32	122.687 (16)	**4.860 (4)**	1581.187 (25)	**632.562 (13)**	1317.968 (17)	**169.390 (6)**	497.828 (11)	**245.109 (7)**	4.359 (5)	**3.437 (3)**
36	**22.860 (9)**	61.578 (11)	220.891 (12)	**131.797 (8)**	395.188 (12)	**115.219 (6)**	294.079 (11)	**80.360 (5)**	**1.422 (3)**	2.563 (3)
40	54.390 (18)	**47.125 (14)**	385.656 (15)	**98.234 (7)**	59.390 (7)	**19.172 (4)**	4905.890 (29)	**949.578 (12)**	23.312 (12)	**1.375 (2)**

Table 5.7: Relative optimality gap of unsolved instances with delayed interdiction, one attacker

Instance	Base	Aug
Data2, Budget 28	5.469%	-
Data 2, Budget 32	8.333%	-
Data2, Budget 36	7.207%	-
Data3, Budget 12	0.562%	-
Data3, Budget 16	2.907%	-
Data3, Budget 24	1.987%	-
Data3, Budget 28	3.471%	-
Data3, Budget 32	3.008%	-
Data3, Budget 36	4.800%	-

Table 5.8: Relative optimality gap of unsolved instances with delayed interdiction, two attackers

Instance	Base	Aug
Data2, Budget 24	3.604%	0.926%
Data2, Budget 28	3.030%	-
Data4, Budget 24	2.326%	3.817%
Data4, Budget 40	5.495%	1.149%

Table 5.9: Relative optimality gap of unsolved instances with upfront interdiction, one attacker

Instance	Base	Aug	State	State Aug
Data2, Budget 40	7.059%	-	2.439%	-
Data3, Budget 24	2.837%	-	2.817%	-
Data3, Budget 28	8.696%	-	4.511%	-
Data3, Budget 32	8.730%	-	3.306%	-
Data3, Budget 36	13.333%	-	-	-
Data4, Budget 40	7.143%	-	-	-

Table 5.10: Relative optimality gap of unsolved instances with upfront interdiction, two attackers

Instance	Base	Aug	State	State Aug
Data2, Budget 24	5.000%	4.00%	-	-
Data 2, Budget 28	4.651%	-	-	-
Data3, Budget 16	5.479%	-	-	-
Data4, Budget 40	1.471 %	-	-	-

Table 5.11: Recommended delayed interdictions on network 1 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	1	2	0	1	2
12	0	1	4	0	2	2
16	0	2	4	1	0	6
20	0	1	8	1	0	8
24	0	1	10	1	0	10
28	0	2	10	1	1	10
32	0	1	14	0	0	16
36	0	2	14	0	0	18
40	0	2	16	0	0	20

Table 5.12: Recommended delayed interdictions on network 3 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	2	0
12	0	3	0	0	0	6
16	0	2	4	1	0	6
20	0	2	6	1	0	8
24	1	2	6	0	0	12
28	2	2	5	1	0	12
32	2	2	7	1	0	14
36	2	2	9	1	0	16
40	1	2	14	1	0	18

Table 5.13: Recommended delayed interdictions on network 1 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	1	2	3	1	0	0	4	2	3
12	0	1	4	2	3	0	0	6	2	3
16	0	1	6	3	1	0	0	8	2	3
20	0	1	8	3	1	0	0	10	2	3
24	0	2	12	2	3	0	0	12	2	4
28	0	1	14	2	3	0	0	14	2	4
32	0	1	14	2	4	0	0	16	2	4
36	0	2	14	2	4	0	0	18	2	4
40	0	3	14	2	4	0	0	20	0	8

Table 5.14: Recommended delayed interdictions on network 3 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP Int Current Vic	MMFNIP 2nd Int Prosp Vic	MMNIP-R 2nd Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	3
12	0	2	2	3	1	0	0	6	2	3
16	0	2	4	3	1	0	0	8	2	4
20	0	2	6	3	1	0	0	10	2	4
24	1	2	6	3	1	0	0	12	2	4
28	0	1	12	2	4	0	0	14	2	4
32	1	3	8	2	4	0	0	16	2	4
36	2	3	7	2	4	0	0	18	1	7
40	1	1	14	2	4	0	0	20	1	7

Table 5.15: Recommended upfront interdictions on network 1 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	0	4
12	1	1	1	1	0	4
16	1	2	1	1	0	6
20	1	3	1	1	0	8
24	0	2	8	1	0	10
28	0	2	10	1	1	10
32	0	2	12	2	2	8
36	1	2	10	2	2	10
40	1	2	12	2	3	10

Table 5.16: Recommended upfront interdictions on network 3 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	0	4
12	0	3	0	1	1	2
16	0	2	4	1	0	6
20	1	2	3	1	0	8
24	1	2	5	1	0	10
28	2	2	5	1	0	12
32	2	3	5	1	0	14
36	2	2	9	1	0	16
40	2	3	9	1	0	18

Table 5.17: Recommended upfront interdictions on network 1 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP Int Current Vic	MMFNIP Int Prosp Vic	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	3
12	0	2	2	2	2	0	0	6	2	3
16	0	2	4	2	3	0	0	8	2	3
20	0	2	6	2	3	0	0	10	2	3
24	0	2	8	2	3	0	0	12	2	4
28	0	2	10	2	3	0	0	14	2	4
32	0	3	10	2	3	0	0	16	2	4
36	0	2	14	2	4	0	0	18	2	4
40	0	3	14	2	4	0	0	20	0	8

Table 5.18: Recommended upfront interdictions on network 3 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP Int Current Vic	MMFNIP Int Prosp Vic	MMNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R Int Current Victim	MP-MFNIP-R Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	3
12	0	2	2	3	1	0	0	6	2	3
16	1	2	1	3	1	0	0	8	2	4
20	2	2	1	3	1	0	0	10	2	4
24	1	2	5	3	1	0	0	12	2	4
28	2	2	5	3	1	0	0	14	2	4
32	1	3	7	2	4	0	0	16	2	4
36	2	3	7	2	4	0	0	18	1	7
40	2	3	9	1	5	0	0	20	1	7

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

This dissertation improved upon operations research tools that are used for the analysis and disruption of illicit networks. New problems in community detection were proposed to assist in the analysis of illicit networks. To help identify missing edges in illicit networks, we proposed a new problem in community detection that adds edges (or weights on edges) to the network to enforce that a known partition must be the partition the maximizes the modularity of the network. To help allocate resources in investigating suspected edges in the network, we proposed a new problem that identifies the most important edges in determining the community structure by removing edges from the network while still preserving the optimal community structure. Exact and heuristic algorithms were proposed to solve these problems, and we demonstrated that our heuristic algorithms provide high quality solutions while significantly reducing the solve time of these problems.

Additionally, new problems in network interdiction were proposed to assist in the disruption of illicit networks. We first introduced the max flow network interdiction problem with restructuring, which allows the defender to add arcs to the network in response to the attacker's interdiction decisions. We modeled this problem as a bilevel mixed integer-linear program, and improved upon column-and-constraint generation to solve this problem. Our results on synthetic city level drug smuggling networks showed that the recommendations from current interdiction models overestimate the impact of the interdictions because they fail to account for the replaceability of lower level participants, such as drug dealers and drug users. These findings mathematically support the findings of criminologists that current drug policy is not effectively disrupting the activities of illicit drug trafficking networks.

We were also among the first works to consider how network interdiction may be applied to human trafficking networks. In order to help address the lack of data in human trafficking, we developed a network generator to produce synthetic domestic sex trafficking networks with a single trafficker. The generator was informed by many sources, including case file analysis, targeted interviews, and a secondary analysis of previously conducted interviews. The networks generated were further validated by qualitative researchers and a survivor-centered advisory group. To apply network interdiction models to these networks, we proposed a novel conceptualization of flow that considers flow as the ability of traffickers

to control their victims. Testing our network interdiction model on networks produced by our generator, we found similar results as with drug trafficking networks; the recruitment of new victims allows for traffickers to recover a significant amount of flow. In this domain, this emphasized the need to disrupt existing sex trafficking networks and disrupt their ability to recruit new victims.

We lastly introduced the multi-period max flow network interdiction problem with restructuring, incorporating a temporal aspect in the network interdiction problem we previously introduced. In this problem, interdiction and restructuring decisions are decided upon upfront, and implemented throughout the time horizon. We additionally proposed a variant of this problem to incorporate a second cooperating attacker that has the ability to prevent the recruit of new participants, modeling how social services would disrupt a sex trafficking network. We proposed modeling-specific augmentations to our column-and-constraint generation algorithm that are able to significantly improve the solve time of our method. By testing our models on networks produced by our generator, we again found that the prevention of recruitment was most effective in disrupting these networks.

There are multiple avenues for future work on each topic. For network interdiction, much work is needed in modeling human trafficking networks before we can responsibly use these models to provide policy recommendations. Conceptualizing flow as control in human trafficking networks is still in its initial stages, so further work into understanding how human trafficking networks operate and can be modeled is needed. A better understanding of how sex trafficking networks operate can be used to improve the accuracy and applicability of our network generator, as well as more accurate modeling of the goals of traffickers and what abilities they have to adapt their operations to respond to disruptions. Another direction of future work incorporates uncertainty into the network. When applying network interdiction models to illicit networks, there is much that is unknown about the network. Combining this idea with a multi-period model, we can devise a model that plans interdiction decisions based on what is known about how the network currently operates and can restructure, and based on what the network could possibly be.

REFERENCES

[1] G. Agarwal and D. Kempe, *Modularity-maximizing graph communities via mathematical programming*, Eur. Phys. J. B, 66 (2008), pp. 409–418.

[2] D. L. Alderson, G. G. Brown, W. M. Carlyle, and R. K. Wood, *Solving defender-attacker-defender models for infrastructure defense*, in 12th INFORMS Computing Society Conference, Monterey, CA, USA, INFORMS, 2011, pp. 28–4–9.

[3] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti, *Column generation algorithms for exact modularity maximization in networks*, Phys. Rev. E, 82 (2010), pp. 1–9.

[4] B. Anthony, J. K. Penrose, S. Jakiel, T. Couture, S. Crowe, M. Fowler, R. Keyhan, K. Sorenson, M. Bradley, and M. A. Badavi, *The typology of modern slavery: Defining sex and labor trafficking in the United States*. Retrieved from `https://polarisproject.org/wp-content/uploads/2019/09/Polaris-Typology-of-Modern-Slavery-1.pdf` (accessed 1 February 2020).

[5] R. Anzoom, R. Nagi, and C. Vogiatzis, *A review of research in illicit supply-chain networks and new directions to thwart them*, IISE Trans., 54 (2021), pp. 134–158.

[6] A. Bahulkar, N. O. Baycik, T. Sharkey, Y. Shen, B. Szymanski, and W. Wallace, *Integrative analytics for detecting and disrupting transnational interdependent criminal smuggling, money, and money-laundering networks*, in 2018 IEEE International Symposium on Technologies for Homeland Security (HST), Boston, MA, USA, IEEE, 2018, pp. 1–6.

[7] A. Bahulkar, B. K. Szymanski, N. O. Baycik, and T. C. Sharkey, *Community detection with edge augmentation in criminal networks*, in 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Barcelona, Spain, IEEE Press, 2018, pp. 1168–1175.

[8] V. Batagelj and A. Mrvar, *Pajek datasets*. Retrieved from `http://vlado.fmf.uni-lj.si/pub/networks/data/esna/default.htm` (accessed 4 June 2020).

[9] N. O. Baycik, T. C. Sharkey, and C. E. Rainwater, *Interdicting layered physical and information flow networks*, IISE Trans., 50 (2018), pp. 316–331.

[10] N. Belles, *Defining sex trafficking*, in Handbook of Sex Trafficking, L. Walker, G. Gaviria, and K. Gopal, eds., Springer, New York, NY, USA, 2018, pp. 3–8.

[11] G. Berlusconi, *Do all the pieces matter? Assessing the reliability of law enforcement data sources for the network analysis of wire taps*, Glob. Crime, 14 (2013), pp. 61–81.

[12] G. Berlusconi, F. Calderoni, N. Parolini, M. Verani, and C. Piccardi, *Link prediction in criminal networks: A tool for criminal intelligence analysis*, PLoS ONE, 11 (2016), p. e0154244.

[13] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*, vol. 6, Athena Scientific, Belmont, MA, USA, 1997.

[14] G. Bichler, A. Malm, and T. Cooper, *Drug supply networks: a systematic review of the organizational structure of illicit drug trade*, Crime Sci., 6 (2017), p. 2.

[15] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, *Fast unfolding of communities in large networks*, J. Stat. Mech: Theory Exp., 2008 (2008), pp. 1–12.

[16] A. Blumstein and A. J. Beck, *Population growth in US prisons, 1980-1996*, Crime Justice, 26 (1999), pp. 17–61.

[17] J. S. Borrero, O. A. Prokopyev, and D. Sauré, *Sequential shortest path interdiction with incomplete information*, Decis. Anal., 13 (2016), pp. 68–98.

[18] ——, *Sequential interdiction with incomplete information and learning*, Oper. Res., 67 (2019), pp. 72–89.

[19] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, *On modularity clustering*, IEEE Trans. Knowl. Data Eng., 20 (2007), pp. 172–188.

[20] D. Bright, J. Koskinen, and A. Malm, *Illicit network dynamics: The formation and evolution of a drug trafficking network*, J. Quant. Criminol., 35 (2019), pp. 237–258.

[21] D. A. Bright, C. Greenhill, and N. Levenkova, *Dismantling criminal networks: Can node attributes play a role?*, in Crime and Networks, C. Morselli, ed., Routledge, London, England, UK, 2013, pp. 160–174.

[22] F. Calderoni, D. Brunetto, and C. Piccardi, *Communities in criminal networks: A case study*, Soc. Networks, 48 (2017), pp. 116–125.

[23] F. Calderoni, S. Catanese, P. De Meo, A. Ficara, and G. Fiumara, *Robust link prediction in criminal networks: A case study of the Sicilian Mafia*, Expert Syst. Appl., 161 (2020), pp. 1–11.

[24] P. Campana, *The structure of human trafficking: Lifting the bonnet on a nigerian transnational network*, Br. J. Criminol., 56 (2016), pp. 68–86.

[25] K. M. Carley, R. Carley, K. Zico, D. Krackhardt, P. V. Singh, and O. Yagan, *Public datasets*. Retrieved from `http://www.casos.cs.cmu.edu/tools/datasets/external/index.php` (accessed 4 June 2020).

[26] A. Carpenter and J. Gates, *The nature and extent of gang involvement in sex trafficking in San Diego County*, US Department of Justice, San Diego, CA, USA, 2016.

[27] J. P. Caulkins, M. Kammer-Kerwick, R. Konrad, K. L. Maass, L. Martin, and T. Sharkey, *A call to the engineering community to address human trafficking*, Bridge (Wash D.C.), 49 (2019), pp. 67–73.

[28] Y. Chandra, *Illicit drug trafficking and financing of terrorism*, J. Def. Stud., 14 (2020), pp. 69–91.

[29] M. Chen, K. Kuzmin, and B. K. Szymanski, *Community detection via maximization of modularity and its variants*, IEEE Trans. Comput. Social Syst., 1 (2014), pp. 46–65.

[30] M. Chen, T. Nguyen, and B. K. Szymanski, *On measuring the quality of a network community structure*, in 2013 International Conference on Social Computing, Alexandria, VA, USA, IEEE, 2013, pp. 122–127.

[31] M. S. Cinar, B. Genc, H. Sever, and V. V. Raghavan, *Analyzing structure of terrorist networks by using graph metrics*, in Proceedings of the 2017 IEEE International Conference on Big Knowledge (ICBK), Hefei, China, IEEE, 2017, pp. 9–16.

[32] E. Cockbain, *Offender and Victim Networks in Human Trafficking*, Routledge, London, England, UK, 2018.

[33] E. Cockbain, K. Bowers, and L. Vernon, *Using law enforcement data in trafficking research*, in The Palgrave International Handbook of Human Trafficking, J. Winterdyk and J. Jones, eds., Palgrave Macmillan, London, England, UK, 2020, pp. 1709–1732.

[34] C. Contardo and J. A. Sefair, *A progressive approximation approach for the exact solution of sparse large-scale binary interdiction games*, INFORMS J. Comput., 34 (2022), pp. 890–908.

[35] K. J. Cormican, D. P. Morton, and R. K. Wood, *Stochastic network interdiction*, Oper. Res., 46 (1998), pp. 184–197.

[36] G. Csardi and T. Nepusz, *The igraph software package for complex network research*, InterJournal, Complex Systems (2006), p. 1695.

[37] S. Cuganesan and D. Lacey, *Developments in public sector performance measurement: A project on producing return on investment metrics for law enforcement*, Financ. Account. Manag., 27 (2011), pp. 458–479.

[38] D. Cunningham, S. Everton, and P. Murphy, *Understanding Dark Networks: A Strategic Framework for the Use of Social Network Analysis*, Rowman & Littlefield, Lanham, Maryland, USA, 2016.

[39] M. Dank, B. Khan, P. Downey, C. Kotonias, D. Mayer, C. Owens, and L. Yu, *Estimating the size and structure of the underground commercial sex economy in eight major US cities*. Retrieved from https://www.ojp.gov/pdffiles1/nij/grants/245295.pdf (accessed 1 February 2020).

[40] M. de Cock and M. Woode, *Profits and Poverty: the Economics of Forced Labour*, International Labour Office, Geneva, Switzerland, 2014.

[41] J. Deeb-Swihart, A. Endert, and A. Bruckman, *Understanding law enforcement strategies and needs for combating human trafficking*, in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, Glasgow, Scotland, UK, Association for Computing Machinery, 2019, pp. 1–14.

[42] K. Descormiers and C. Morselli, *Alliances, conflicts, and contradictions in Montreal's street gang landscape*, Int. Crim. Justice Rev., 21 (2011), pp. 297–314.

[43] G. L. Dimas, R. A. Konrad, K. L. Maass, and A. C. Trapp, *A Survey of Operations Research and Analytics Literature Related to Anti-Human Trafficking*. preprint, arXiv:2103.16476 [cs.CY], 2021.

[44] Drug Enforcement Administration, *National drug threat assessment*. Retrieved from https://www.dea.gov/sites/default/files/2020-01/2019-NDTA-final-01-14-2020_Low_Web-DIR-007-20_2019.pdf (accessed 1 February 2020).

[45] A. Dubrawski, K. Miller, M. Barnes, B. Boecking, and E. Kennedy, *Leveraging publicly available data to discern patterns of human-trafficking activity*, J. Hum. Traffick., 1 (2015), pp. 65–85.

[46] A. Elsisy, A. Mandviwalla, B. Szymanski, and T. Sharkey, *A Synthetic Network Generator for Covert Network Analytics*. preprint, arXiv:2008.04445 [cs.SI], 2020.

[47] A. Farrell and I. de Vries, *Measuring the nature and prevalence of human trafficking*, in The Palgrave International Handbook of Human Trafficking, J. Winterdyk and J. Jones, eds., Palgrave Macmillan, London, England, UK, 2020, pp. 147–162.

[48] L. Fedina, *Use and misuse of research in books on sex trafficking: Implications for interdisciplinary researchers, practitioners, and advocates*, Trauma Violence Abuse, 16 (2015), pp. 188–198.

[49] M. Fischetti, I. Ljubic, M. Monaci, and M. Sinnl, *Interdiction games and monotonicity, with application to knapsack problems*, INFORMS J. Comput., 31 (2019), pp. 390–410.

[50] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl, *A new general-purpose algorithm for mixed-integer bilevel linear programs*, Oper. Res., 65 (2017), pp. 1615–1637.

[51] L. Gerassi, T. Edmond, and A. Nichols, *Design strategies from sexual exploitation and sex work studies among women and girls: Methodological considerations in a hidden and vulnerable population*, Action Res. (Lond), 15 (2017), pp. 161–176.

[52] J. Gill and J. R. Freeman, *Dynamic elicited priors for updating covert networks*, Network Sci., 1 (2013), pp. 68–94.

[53] R. Goldstein, M. W. Sances, and H. Y. You, *Exploitative revenues, law enforcement, and the quality of government service*, Urban Aff. Rev., 56 (2020), pp. 5–31.

[54] T. U. Grund and J. A. Densley, *Ethnic homophily and triad closure: Mapping internal gang structure using exponential random graph models*, J. Contemp. Crim. Justice, 31 (2015), pp. 354–370.

[55] M. D. Holmes, B. W. Smith, A. B. Freng, and E. A. Muñoz, *Minority threat, crime control, and police resource allocation in the Southwestern United States*, Crime Delinq., 54 (2008), pp. 128–152.

[56] T. Holzmann and J. C. Smith, *Shortest path interdiction problem with arc improvement recourse: A multiobjective approach*, Nav. Res. Logist., 66 (2019), pp. 230–252.

[57] IBM, *User's manual for CPLEX, version 20.1.* Retrieved from `https://www.ibm.com/docs/en/icos/20.1.0?topic=cplex-users-manual` (accessed 23 April 2019).

[58] Z. Jabarzare, H. Zolfagharinia, and M. Najafi, *Dynamic interdiction networks with applications in illicit supply chains*, Omega, 96 (2020), pp. 1–22.

[59] U. Janjarassuk and J. Linderoth, *Reformulation and sampling to solve a stochastic network interdiction problem*, Networks, 52 (2008), pp. 120–132.

[60] R. J. Kelly, J. Maghan, and J. Serio, *Illicit Trafficking: A Reference Handbook*, ABC-CLIO, Santa Barbara, CA, USA, 2005.

[61] L. W. Kennedy, J. M. Caplan, and E. Piza, *Risk clusters, hotspots, and spatial intelligence: Risk terrain modeling as an algorithm for police resource allocation strategies*, J. Quant. Criminol., 27 (2011), pp. 339–362.

[62] B. B. Keskin, G. J. Bott, and N. K. Freeman, *Cracking sex trafficking: Data analysis, pattern recognition, and path prediction*, Prod. Oper. Manag., 30 (2021), pp. 1110–1135.

[63] S. S. Ketkov and O. A. Prokopyev, *On greedy and strategic evaders in sequential interdiction settings with incomplete information*, Omega, 92 (2020), pp. 1–15.

[64] D. E. Knuth, *The Stanford GraphBase: A Platform for Combinatorial Computing*, ACM Press, New York, NY, USA, 1993.

[65] R. Konrad, A. Trapp, and K. L. Maass, *Human trafficking analysis*, OR/MS Today, 44 (2017), pp. 30–33.

[66] R. A. Konrad, A. C. Trapp, T. M. Palmbach, and J. S. Blom, *Overcoming human trafficking via operations research and analytics: Opportunities for methods, models, and applications*, Eur. J. Oper. Res., 259 (2017), pp. 733–745.

[67] D. Kosmas, T. C. Sharkey, J. E. Mitchell, K. L. Maass, and L. Martin, *Interdicting Restructuring Networks with Applications in Illicit Trafficking.* preprint, arXiv:2011.07093 [math.OC], 2020.

[68] X. Lei, S. Shen, and Y. Song, *Stochastic maximum flow interdiction problems under heterogeneous risk preferences*, Comp. Oper. Res., 90 (2018), pp. 97–109.

[69] L. Li, O. Simek, A. Lai, M. Daggett, C. K. Dagli, and C. Jones, *Detection and characterization of human trafficking networks using unsupervised scalable text template matching*, in 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, IEEE, 2018, pp. 3111–3120.

[70] G. R. Lotrecchiano and S. Misra, *Transdisciplinary knowledge producing teams: Toward a complex systems perspective*, Inf. Sci., 21 (2018), pp. 51–74.

[71] L. Lozano and J. C. Smith, *A backward sampling framework for interdiction problems with fortification*, INFORMS J. Comput., 29 (2017), pp. 123–139.

[72] ——, *A value-function-based exact approach for the bilevel mixed-integer programming problem*, Oper. Res., 65 (2017), pp. 768–786.

[73] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, *The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations*, Behav. Ecol. Sociobiol., 54 (2003), pp. 396–405.

[74] N. R. Magliocca, K. McSweeney, S. E. Sesnie, E. Tellman, J. A. Devine, E. A. Nielsen, Z. Pearson, and D. J. Wrathall, *Modeling cocaine traffickers and counterdrug interdiction forces as a complex adaptive system*, Proc. Natl. Acad. Sci. U.S.A., 116 (2019), pp. 7784–7792.

[75] A. Malaviya, C. Rainwater, and T. C. Sharkey, *Multi-period network interdiction problems with applications to city-level drug enforcement*, IIE Trans., 44 (2012), pp. 368–380.

[76] A. Malm and G. Bichler, *Networks of collaborating criminals: Assessing the structural vulnerability of drug markets*, J. Res. Crime Delinq., 48 (2011), pp. 271–297.

[77] M. Mancuso, *Not all madams have a central role: Analysis of a nigerian sex trafficking network*, Trends Organ. Crime, 17 (2014), pp. 66–88.

[78] A. Marcus, A. Horning, R. Curtis, J. Sanson, and E. Thompson, *Conflict and agency among sex workers and pimps: A closer look at domestic minor sex trafficking*, Ann. Am. Acad. Pol. Soc. Sci., 653 (2014), pp. 225–246.

[79] L. Martin and R. Lotspeich, *A benefit-cost framework for early intervention to prevent sex trading*, J. Benefit-Cost Anal., 5 (2014), pp. 43–87.

[80] L. Martin, C. Melander, H. Karnik, and C. Nakamura, *Mapping the demand: Sex buyers in the state of Minnesota*. Retrieved from `https://conservancy.umn.edu/bitstream/handle/11299/226521/MappingtheDemand-FullReport%20-%20FINAL%20July%2031%202017.pdf?sequence=1` (accessed 1 February 2020).

[81] L. E. Martin, A. Pierce, S. Peyton, A. I. Gabilondo, and G. Tulpule, *Mapping the market for sex with minor trafficked girls in Minneapolis: Structures functions, and patterns*. Community Report, University of Minnesota, Minneapolis, MN, 2014.

[82] V. Martínez, F. Berzal, and J.-C. Cubero, *A survey of link prediction in complex networks*, ACM Comput. Surv., 49 (2016), pp. 1–33.

[83] M. Mayorga, L. Tateosian, G. Velasquez, R. Amindarbari, and S. Caltagirone, *Countering human trafficking using ISE/OR techniques*, in Emerging Frontiers in Industrial and Systems Engineering, H. B. Nembhard, E. A. Cudney, and K. M. Coperich, eds., CRC Press, Boca Raton, FL, USA, 2019, pp. 237–257.

[84] G. P. McCormick, *Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems*, Math. Prog., 10 (1976), pp. 147–175.

[85] C. Melander, L. Martin, and K. L. Maass, *Conceptualizing operations that traffic people for sexual exploitation within the United States: Relationship networks within the broader socioecological context*. manuscript, 2022.

[86] J. H. Michael and J. G. Massey, *Modeling the communication network in a sawmill*, For. Prod. J., 47 (1997), pp. 25–30.

[87] P. Mohanty, *A Computational Approach to Identify Covertness and Collusion in Social Networks*, PhD thesis, University of Minnesota, Minneapolis, MN, 2020.

[88] W. T. Morris, *On the art of modeling*, Manage. Sci., 13 (1967), pp. 707–717.

[89] C. Morselli, *Inside Criminal Networks*, Springer, New York, NY, USA, 2009.

[90] D. P. Morton, F. Pan, and K. J. Saeger, *Models for nuclear smuggling interdiction*, IIE Trans., 39 (2007), pp. 3–14.

[91] M. Natarajan, *Understanding the structure of a drug trafficking organization: A conversational analysis*, Crime Prev. Stud., 11 (2000), pp. 273–298.

[92] ——, *Understanding the structure of a large heroin distribution network: A quantitative analysis of qualitative data*, J. Quant. Criminol., 22 (2006), pp. 171–192.

[93] M. E. Newman, *Random graphs as models of networks*, in Handbook of Graphs and Networks: From the Genome to the Internet, S. Bornholdt and H. G. Schuster, eds., Wiley Online Library, Hoboken, NJ, USA, 2003, pp. 35–68.

[94] ——, *Fast algorithm for detecting community structure in networks*, Phys. Rev. E, 69 (2004), pp. 1–5.

[95] ——, *Equivalence between modularity optimization and maximum likelihood methods for community detection*, Phys. Rev. E, 94 (2016), pp. 1–8.

[96] M. E. Newman and M. Girvan, *Finding and evaluating community structure in networks*, Phys. Rev. E, 69 (2004), pp. 1–15.

[97] J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio, *Orbital branching*, Math. Prog., 126 (2011), pp. 147–178.

[98] J. H. Owen and S. Mehrotra, *On the value of binary expansions for general mixed-integer linear programs*, Oper. Res., 50 (2002), pp. 810–819.

[99] S. Pemmaraju and S. Skiena, *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica®*, Cambridge University Press, Cambridge, England, UK, 2003.

[100] G. Potter, *Exploring retail-level drug distribution: Social supply, 'real' dealers and the user/dealer interface*, in Old and New Policies, Theories, Research Methods and Drug Users Across Europe, Pabst Science Publishers, Lengerich, Germany, 2009, pp. 50–74.

[101] K. M. Preble, *Under their 'control': Perceptions of traffickers' power and coercion among international female trafficking survivors during exploitation*, Vict. Offenders, 14 (2019), pp. 199–221.

[102] M. A. Rad and H. T. Kakhki, *Maximum dynamic network flow interdiction problem: New formulation and solution procedures*, Comput. Ind. Eng., 65 (2013), pp. 531–536.

[103] C. Rhodes and P. Jones, *Inferring missing links in partially observed social networks*, J. Oper. Res. Soc., 60 (2009), pp. 1373–1383.

[104] C. J. Rhodes and E. Keefe, *Social network topology: a bayesian approach*, J. Oper. Res. soc., 58 (2007), pp. 1605–1611.

[105] D. Roe-Sepowitz, *A six-year analysis of sex traffickers of minors: Exploring characteristics and sex trafficking patterns*, J. Hum. Behav. Soc. Environ., 29 (2019), pp. 608–629.

[106] J. A. Sefair and J. C. Smith, *Dynamic shortest-path interdiction*, Networks, 68 (2016), pp. 315–330.

[107] B. Serrano and T. Vidal, *Community Detection in the Stochastic Block Model by Mixed Integer Programming*. preprint, arXiv:2101.12336 [cs.SI], 2021.

[108] T. C. Sharkey, K. Barrick, A. Farrell, K. L. Maass, L. Martin, and Y. Song, *Better together: A transdisciplinary approach to disrupt human trafficking*, ISE Magazine, 53 (2021), pp. 34–39.

[109] Y. Shen, T. C. Sharkey, B. K. Szymanski, and W. A. Wallace, *Interdicting interdependent contraband smuggling, money and money laundering networks*, Socio-Econ. Plan. Sci., 78 (2021), pp. 1–28.

[110] J. C. Smith and Y. Song, *A survey of network interdiction models and algorithms*, Eur. J. Oper. Res., 283 (2020), pp. 797–811.

[111] M. Soleimani-Alyar and A. Ghaffari-Hadigheh, *Solving multi-period interdiction via generalized bender's decomposition*, Acta Math. Appl. Sinica, 33 (2017), pp. 633–644.

[112] ——, *Dynamic network interdiction problem with uncertain data*, Int. J. Uncertainty Fuzziness Knowledge Based Syst., 26 (2018), pp. 327–342.

[113] T. Spapens, *Interaction between criminal groups and law enforcement: The case of ecstasy in the Netherlands*, Glob. Crime, 12 (2011), pp. 19–40.

[114] M. K. Sparrow, *The application of network analysis to criminal intelligence: An assessment of the prospects*, Soc. Networks, 13 (1991), pp. 251–274.

[115] H. v. Stackelberg, *Theory of the Market Economy*, Oxford University Press, Oxford, England, UK, 1952.

[116] S. J. Strang, *Network analysis in criminal intelligence*, in Networks and Network Analysis for Defence and Security, A. J. Matsys, ed., Springer, New York, NY, USA, 2014, pp. 1–26.

[117] K. Taha, *Static and dynamic community detection methods that optimize a specific objective function: A survey and experimental evaluation*, IEEE Access, 8 (2020), pp. 98330–98358.

[118] S. Tahernejad, T. K. Ralphs, and S. T. DeNegre, *A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation*, Math. Program. Comput., 12 (2020), pp. 1–40.

[119] B. Tezcan and K. L. Maass, *Human trafficking interdiction with decision dependent success*. preprint, engrXiv 1068, 2020.

[120] R. C. Van der Hulst, *Introduction to Social Network Analysis (SNA) as an investigative tool*, Trends Organ. Crime, 12 (2009), pp. 101–121.

[121] D. J. Watts and S. H. Strogatz, *Collective dynamics of 'small-world'networks*, Nature, 393 (1998), pp. 440–442.

[122] R. Weitzer, *New directions in research on human trafficking*, Ann. Am. Acad. Pol. Soc. Sci., 653 (2014), pp. 6–24.

[123] A. White, S. Guikema, and B. Carr, *Why are you here? Modeling illicit massage business location characteristics with machine learning*, J. Hum. Traffick., (2021), pp. 1–21. https://doi.org/10.1080/23322705.2021.1982238.

[124] T. R. Willemain, *Insights on modeling from a dozen experts*, Oper. Res., 42 (1994), pp. 213–222.

[125] ——, *Model formulation: What experts think about and when*, Oper. Res., 43 (1995), pp. 916–932.

[126] C. Williamson and M. Prior, *Domestic minor sex trafficking: A network of underground players in the midwest*, J. Child Adolesc. Trauma, 2 (2009), pp. 46–61.

[127] R. Wood, *Deterministic network interdiction*, Math. Comput. Model., 17 (1993), pp. 1–18.

[128] X. Xie and F. Aros-Vera, *An interdependent network interdiction model for disrupting sex trafficking networks*, Prod. Oper. Manag., (2022). https://doi.org/10.1111/poms.13713.

[129] D. Yue, J. Gao, B. Zeng, and F. You, *A projection-based reformulation and decomposition algorithm for global optimization of a class of mixed integer bilevel linear programs*, J. Global Optim., 73 (2019), pp. 27–57.

[130] W. W. Zachary, *An information flow model for conflict and fission in small groups*, J. Anthropol. Res., 33 (1977), pp. 452–473.

[131] B. Zeng and Y. An, *Solving bilevel mixed integer program by reformulations and decomposition*. Technical Report, University of Southern Florida, Tampa, FL, 2014.

APPENDIX A
PROOFS OF MODULARITY PROPERTIES

A.1 Change in Modularity from Including an Edge Within a Cluster

Proof. Consider adding edge $e = (u, v)$ where $u, v \in C_1$. Let z^e be the vector such that $z^e_e = 1$ and $z^e_{e'} = 0$ for $e' \neq e$. We compute the difference $(Q_T(z^e) - Q_T(0)$. To simplify fractions, we multiply by $(2m)^2(2m+2)^2$.

$$
\begin{aligned}
&(2m)^2(2m+2)^2(Q_T(z^e) - Q_T(0)) \qquad (A.1)\\
&= (2m)^2(2m+2)A_{uu} - (2m)^2(d_u+1)^2 - (2m)(2m+2)^2A_{uu} + (2m+2)^2d_u^2\\
&+(2m)^2(2m+2)A_{vv} - (2m)^2(d_v+1)^2 - (2m)(2m+2)^2A_{vv} + (2m+2)^2d_v^2\\
&+2\left((2m)^2(2m+2)A_{uv} + (2m)^2(2m+2) - (2m)^2(d_u+1)(d_v+1)\right)\\
&+\left(-(2m)(2m+2)^2A_{uv} + (2m+2)^2d_ud_v\right)\\
&+2\left(\sum_{u,v\neq x\in C_1}\left((2m)^2(2m+2)A_{ux} - (2m)^2(d_u+1)d_x - (2m)(2m+2)^2A_{ux} + (2m+2)^2d_ud_x\right)\right) \qquad (A.2)\\
&+2\left(\sum_{u,v\neq y\in C_1}\left((2m)^2(2m+2)A_{vy} - (2m)^2(d_v+1)d_y - (2m)(2m+2)^2A_{vy} + (2m+2)^2d_vd_y\right)\right)\\
&+\sum_{u,v\neq x,y\in C_1}\left((2m)^2(2m+2)A_{xy} - (2m)^2d_xd_y - (2m)(2m+2)^2A_{xy} + (2m+2)^2d_xd_y\right)\\
&+\sum_{C_i\in T,i\neq 1}\sum_{x,y\in C_i}\left((2m)^2(2m+2)A_{xy} - (2m)^2d_xd_y - (2m)(2m+2)^2A_{xy} + (2m+2)^2d_xd_y\right)
\end{aligned}
$$

$$
\begin{aligned}
&= (-8m^2-8m)A_{uu} - 8m^2d_u - 4m^2 + 8md_u^2 + 4d_u^2\\
&+(-8m^2-8m)A_{vv} - 8m^2d_v - 4m^2 + 8md_v^2 + 4d_v^2\\
&+2\left((-8m^2-8m)A_{uv} + 8m^3 + 8m^2 - (4m^2)(d_ud_v + d_u + d_v + 1) + (4m^2+8m+4)d_ud_v\right)\\
&+2\left(\sum_{u,v\neq x\in C_1}\left((-8m^2-8m)A_{ux} - 4m^2(d_ud_x + d_x) + (4m^2+8m+4)d_ud_x\right)\right) \qquad (A.3)\\
&+2\left(\sum_{u,v\neq y\in C_1}\left((-8m^2-8m)A_{vy} - 4m^2(d_vd_y + d_y) + (4m^2+8m+4)d_vd_y\right)\right)\\
&+\sum_{u,v\neq x,y\in C_1}\left((-8m^2-8m)A_{xy} + (8m+4)d_xd_y\right)\\
&+\sum_{C_i\in T,i\neq 1}\sum_{x,y\in C_i}\left((-8m^2-8m)A_{xy} + (8m+4)d_xd_y\right)
\end{aligned}
$$

$$
\begin{aligned}
&= (-8m^2 - 8m)A_{uu} + (8m+4)d_u^2 - 8m^2 d_u - 4m^2 \\
&+(-8m^2 - 8m)A_{vv} + (8m+4)d_v^2 - 8m^2 d_v - 4m^2 \\
&+2\left((-8m^2 - 8m)A_{uv} + (8m+4)d_u d_v + 8m^3 + 4m^2 - 4m^2 d_u - 4m^2 d_v\right) \\
&+2\left(\sum_{u,v\neq x\in C_1}\left((-8m^2 - 8m)A_{ux} + (8m+4)d_u d_x - 4m^2 d_x\right)\right) \\
&+2\left(\sum_{u,v\neq y\in C_1}\left((-8m^2 - 8m)A_{vy} + (8m+4)d_v d_y - 4m^2 dy\right)\right) \\
&+\sum_{u,v\neq x,y\in C_1}\left((-8m^2 - 8m)A_{xy} + (8m+4)d_x d_y\right) \\
&+\sum_{C_i\in T, i\neq 1}\sum_{x,y\in C_i}\left((-8m^2 - 8m)A_{xy} + (8m+4)d_x d_y\right)
\end{aligned}
\tag{A.4}
$$

$$
= 16m^3 - \sum_{x\in C_1} 16m^2 d_x + \sum_{C_i\in T}\sum_{x,y\in C_i}\left((-8m^2 - 8m)A_{xy} + (8m+4)d_x d_y\right) \tag{A.5}
$$

Thus, the increase in modularity by adding (u, v) is

$$
\frac{16m^3 - \sum_{x\in C_1} 16m^2 d_x + \sum_{C_i\in T}\sum_{x,y\in C_i}\left((-8m^2 - 8m)A_{xy} + (8m+4)d_x d_y\right)}{(2m)^2(2m+2)^2}. \tag{A.6}
$$

□

A.2 Change in Modularity from Including an Edge Between Clusters

Proof. Consider adding edge $e = (u, v)$ where $u \in C_1$ and $v \in C_2$. Let z^e be the vector such that $z^e_e = 1$ and $z^e_{e'} = 0$ for $e' \neq e$. We compute the difference $(Q_T(z^e) - Q_T(0)$. We simplify fractions by multiplying by $(2m)^2(2m+2)^2$.

$$(2m)^2(2m+2)^2(Q_T(z^e) - Q_T(0)) \tag{A.7}$$

$$\begin{aligned}
&= (2m)^2(2m+2)A_{uu} - (2m)^2(d_u+1)^2 - (2m)(2m+2)^2A_{uu} + (2m+2)^2d_u^2 \\
&+2\left(\sum_{u\neq x\in C_1}\left((2m)^2(2m+2)A_{ux} - (2m)^2(d_u+1)d_x - (2m)(2m+2)^2A_{ux} + (2m+2)^2d_ud_x\right)\right) \\
&\quad + \sum_{u\neq x,y\in C_1}\left((2m)^2(2m+2)A_{xy} - (2m)^2d_xd_y - (2m)(2m+2)^2A_{xy} + (2m+2)^2d_xd_y\right) \\
&\quad +(2m)^2(2m+2)A_{vv} - (2m)^2(d_v+1)^2 - (2m)(2m+2)^2A_{vv} + (2m+2)^2d_v^2 \\
&+2\left(\sum_{v\neq y\in C_2}\left((2m)^2(2m+2)A_{vy} - (2m)^2(d_v+1)d_y - (2m)(2m+2)^2A_{vy} + (2m+2)^2d_vd_y\right)\right) \\
&\quad + \sum_{v\neq x,y\in C_2}\left((2m)^2(2m+2)A_{xy} - (2m)^2d_xd_y - (2m)(2m+2)^2A_{xy} + (2m+2)^2d_xd_y\right) \\
&+ \sum_{C_i\in T, i\notin\{1,2\}}\sum_{x,y\in C_i}\left((2m)^2(2m+2)A_{xy} - (2m)^2d_xd_y - (2m)(2m+2)^2A_{xy} + (2m+2)^2d_xd_y\right)
\end{aligned} \tag{A.8}$$

$$\begin{aligned}
&= (-8m^2-8m)A_{uu} - 8m^2d_u - 4m^2 + 8md_u^2 + 4d_u^2 \\
&+2\left(\sum_{u\neq x\in C_1}\left((-8m^2-8m)A_{ux} - 4m^2(d_ud_x+d_x) + (4m^2+8m+4)d_ud_x\right)\right) \\
&\quad + \sum_{u\neq x,y\in C_1}\left((-8m^2-8m)A_{xy} + (8m+4)d_xd_y\right) \\
&\quad +(-8m^2-8m)A_{vv} - 8m^2d_v - 4m^2 + 8md_v^2 + 4d_v^2 \\
&+2\left(\sum_{v\neq y\in C_2}\left((-8m^2-8m)A_{vy} - 4m^2(d_vd_y+d_y) + (4m^2+8m+4)d_vd_y\right)\right) \\
&\quad + \sum_{v\neq x,y\in C_2}\left((-8m^2-8m)A_{xy} + (8m+4)d_xd_y\right) \\
&\quad + \sum_{C_i\in T, i\notin\{1,2\}}\sum_{x,y\in C_i}\left((-8m^2-8m)A_{xy} + (8m+4)d_xd_y\right)
\end{aligned} \tag{A.9}$$

$$
\begin{aligned}
&= (-8m^2 - 8m)A_{uu} + (8m+4)d_u^2 - 8m^2 d_u - 4m^2 \\
&+2\left(\sum_{u\neq x\in C_1}\left((-8m^2-8m)A_{ux} + (8m+4)d_u d_x - 4m^2 d_x\right)\right) \\
&+\sum_{u\neq x,y\in C_1}\left((-8m^2-8m)A_{xy} + (8m+4)d_x d_y\right) \\
&+(-8m^2-8m)A_{vv} + (8m+4)d_v^2 - 8m^2 d_v - 4m^2 \\
&+2\left(\sum_{v\neq y\in C_2}\left((-8m^2-8m)A_{vy} + (8m+4)d_v d_y - 4m^2 dy\right)\right) \\
&+\sum_{v\neq x,y\in C_2}\left((-8m^2-8m)A_{xy} + (8m+4)d_x d_y\right) \\
&+\sum_{C_i\in T, i\notin\{1,2\}}\sum_{x,y\in C_i}\left((-8m^2-8m)A_{xy} + (8m+4)d_x d_y\right)
\end{aligned} \tag{A.10}
$$

$$
= -8m^2 - \sum_{x\in C_1} 8m^2 d_x - \sum_{y\in C_2} 8m^2 d_y + \sum_{C_i\in T}\sum_{x,y\in C_i}\left((-8m^2-8m)A_{xy} + (8m+4)d_x d_y\right) \tag{A.11}
$$

Thus, the increase in modularity by adding (u, v) is

$$
\frac{-8m^2 - \sum_{x\in C_1} 8m^2 d_x - \sum_{y\in C_2} 8m^2 d_y + \sum_{C_i\in T}\sum_{x,y\in C_i}\left((-8m^2-8m)A_{xy} + (8m+4)d_x d_y\right)}{(2m)^2(2m+2)^2}. \tag{A.12}
$$

□

APPENDIX B
COMPARISON OF MASTER AND SUBPROBLEM SOLVE TIMES IN MODULARITY MAXIMIZATION

Table B.1: Comparison of run times (in seconds) with default CPLEX symmetry breaking

Network	Master (IP)	Sub (IP)	Master (IP+)	Sub (IP+)
Sawmill	288.89	10.09	3050.45	5.78
Dolphins	>7200	27.81	>7200	25.24
Karate	514.06	12.56	51.39	8.30
Les Mis	83.01	386.96	20.41	166.31
Ciel	1.13	0.20	1.09	0.19
Caviar6	115.45	7.44	27.92	4.92
Rhodes	>7200	5.88	>7200	4.20
Montreal Gangs	>7200	24.64	>7200	103.38
Italian Gangs	5.06	17.02	4.53	13.33
London Gangs	24.13	31.36	12.42	16.88

Table B.2: Comparison of run times (in seconds) with maximum CPLEX symmetry breaking

Network	Master (IP)	Sub (IP)	Master (IP+)	Sub (IP+)
Sawmill	261.12	9.98	3265.61	5.75
Dolphins	>7200	15.64	>7200	67.66
Karate	432.62	12.52	51.31	7.94
Les Mis	83.41	366.24	30.84	168.44
Ciel	1.11	0.20	1.13	0.20
Caviar6	112.99	7.59	27.22	4.95
Rhodes	>7200	5.53	>7200	1.94
Montreal Gangs	>7200	25.14	>7200	75.36
Italian Gangs	4.84	17.36	4.81	14.75
London Gangs	24.11	30.55	14.09	18.64

APPENDIX C
PROOFS OF MFNIP-R LEMMAS

Proof of Lemma 3:

Proof. Consider arc (i, j), where $i \in \bar{U}$. Since $i \in \bar{U}$, i is not reachable from α in D_f, and thus there is no path from α to i in D_f. Note that including (i, j) in our network introduces the same arc (i, j) in the auxiliary network D_f. Since there is no path from α to i in D_f, including (i, j) will not complete a path from α to ω in D_f. Thus, restructuring arc (i, j) will not increase the value of the flow. □

Proof of Lemma 4:

Proof. Consider arc (i, j), where $i \in U$ and $j \in U$. Since $i \in U$, i is reachable from α in D_f, and thus there exists a path from α to i in D_f. Likewise, since $j \in U$, then j is reachable from α in D_f, and thus there exists a path from α to j in D_f. This implies that there cannot exist a path from j to ω in D_f. If there were such a path, then we would have a path from α to ω, and thus the current flow is not optimal. Since there is no path from j to ω in D_f, including arc (i, j) will not complete a path from α to ω in D_f. Thus, the value of the flow will not increase if this arc is introduced. □

Proof of Lemma 5:

Proof. Consider arc (i, j), where $i \in U$ and $j \in \bar{U}$, and there is no path from j to ω in D_f. Since $i \in U$, there exists a path from α to i in D_f, and thus there is no path from i to ω in D_f, else the original flow is not optimal. Since there is no path from j to ω in D_f, including arc (i, j) will not complete a path from α to ω in D_f. Thus, the value of the flow will not increase. □

APPENDIX D
MFNIP-R COMPUTATIONAL RESULTS

Entries marked with * are instances where the method did not solve within the two hour time limit. We instead list the upper bound, as this is the best known bilevel feasible solution, and the time of last completed iteration.

Table D.1: Optimal flows for different base data sets with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
Base	116.77137	111.89403	114.435134	110.546185	118.280546
50	77.2257	49.64045	71.548639	63.32806	81.92476
60	64.72307	40.59544	63.378588	53.63989	71.12085
70	55.44162	32.54936	54.32644	44.69965	62.40129
80	46.65828	21.85086	46.196279	36.37513	53.86366
90	36.38575	13.76413	38.230575	27.284	43.39701
100	27.08376	6.3368	29.14647	18.14417	36.78273
110	19.46085	0	21.07597	11.48059	29.61927
120	12.79303	0	14.95574	7.04445	21.7783
130	6.14819	0	9.099645	1.90682	14.82776
140	0	0	2.844335	0	9.484119

Table D.2: Run time in seconds (number of plans visited) for different base data sets with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
50	110.875 (3)	10.938 (3)	25.35 (3)	18.86 (3)	139.688 (3)
60	18.844 (3)	5.672 (2)	521.078 (8)	23.75 (3)	70.422 (2)
70	12.781 (2)	22.968 (3)	449.516 (7)	51.765 (4)	177.031 (3)
80	48.313 (3)	47.078 (3)	1316.953 (7)	104.438 (4)	170.422 (3)
90	45.203 (2)	122.359 (3)	688.641 (5)	139.515 (4)	56.063 (2)
100	61.188 (2)	95.031 (3)	69.656 (2)	722.641 (5)	210.328 (3)
110	80.078 (2)	3.219 (2)	118.578 (2)	409.906 (4)	312.594 (3)
120	378.562 (3)	0.156 (1)	90.485 (2)	968.485 (6)	97.468 (2)
130	636.907 (3)	0.219 (1)	1893.594 (7)	537.984 (6)	219.719 (3)
140	14.343 (2)	0.141 (1)	4832.516 (12)	62.437 (4)	758.61 (5)

Table D.3: Optimal flows for different data sets with recruitment with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
Base	116.77137	111.89403	114.435134	110.546185	118.280546
50	81.30466	63.862729	77.551249	69.72654	83.62605
60	72.282593	53.256709	67.19034	61.85482	75.2244
70	64.070003	44.619742	59.29457	54.50484	67.24456
80	53.155684	36.102209	53.44689	47.1921	59.20083
90	43.709545	27.680039	45.55112	38.78783*	51.62168
100	36.09047	21.568024	36.29383	30.01219	44.75593
110	26.64771	12.66219	28.37822	22.45318	39.54587*
120	18.16082	4.58978	20.48245	14.29821	36.16806*
130	7.3016	0	15.68868	9.682533	26.62906
140	0	0	6.92709	3.27475	16.56582

Table D.4: Run time in seconds (number of plans visited) for different data sets with recruitment with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
50	55.5 (3)	37.75 (4)	211.891 (6)	70.407 (4)	161.484 (5)
60	70.172 (4)	48.11 (4)	62.687 (4)	37.453 (2)	166.766 (4)
70	344.578 (6)	45.953 (3)	114.157 (4)	129.265 (4)	520.172 (5)
80	575.875 (8)	163.906 (4)	1421.5 (6)	1463.5 (7)	2379.765 (8)
90	1799.312 (5)	179.922 (5)	3364.844 (6)	7200 (8)*	1892.297 (7)
100	1543.235 (5)	962.062 (5)	720.406 (4)	2831.515 (6)	2406.187 (7)
110	1288.687 (5)	260.438 (4)	705.375 (4)	3586.438 (7)	7200 (8)*
120	2373.344 (6)	527.531 (5)	851 (4)	718.875 (3)	7200 (9)*
130	161.642 (2)	67.265 (4)	595.157 (4)	4260 (8)	644.422 (4)
140	28.375 (2)	0.141 (1)	249.781 (3)	714.016 (5)	457.531 (3)

Table D.5: Optimal flows for different data sets with organizational restructuring with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
Base	116.77137	111.89403	114.435134	110.546185	118.280546
50	77.2257	49.64045	71.548639	63.32806	81.92476
60	64.72307	40.59544	63.378588	53.63989	71.12085
70	55.44162	32.54936	54.32644	44.69965	62.40129
80	46.65728	21.85086	46.196279	36.37513	53.86366
90	36.38575	13.76413	38.230575	27.284	43.397010
100	27.08376	6.33368	30.637375	18.14417	36.78273
110	19.46085	0	23.122495	11.48059	29.61927
120	12.79303	0	15.856735	7.04445	21.7783
130	6.14819	0	9.099645	1.90682	14.82776
140	0	0	2.844335	0	9.484119

Table D.6: Run time in seconds (number of plans visited) for different data sets with organizational restructuring with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
50	93.812 (3)	10.797 (3)	30.39 (3)	14.938 (3)	91.469 (3)
60	18.438 (3)	6.453 (2)	735.063 (8)	18.828 (3)	55.578 (2)
70	11.328 (2)	27.547 (3)	448.094 (7)	50.515 (4)	178.766 (3)
80	46.109 (3)	49.735 (3)	1185.343 (7)	116.766 (4)	168.125 (3)
90	48.031 (2)	119.812 (3)	644.937 (5)	162.078 (4)	56.422 (2)
100	49.984 (2)	114.078 (3)	1669.125 (7)	706.016 (5)	234.938 (3)
110	80.125 (2)	4.688 (2)	1906.25 (7)	407.812 (4)	338.156 (3)
120	424.891 (3)	0.125 (1)	1782.188 (7)	981.297 (6)	131.39 (2)
130	150.859 (2)	2.063 (3)	3860.046 (7)	437.812 (6)	564.625 (3)
140	197.172 (3)	0.125 (1)	3247.78 (10)	17.328 (4)	892.281 (4)

Table D.7: Optimal flows for different base data sets with interdicting leadership with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
Base	116.77137	111.89403	114.435134	110.546185	118.280546
50	87.99789	69.30883	85.118192	71.85268	-
60	74.99896	60.12165	66.37184	60.58043	101.51818
70	64.18631	49.64045	54.87111	51.50233	85.87106
80	55.52443	32.50457	47.2207	43.40857	75.35971
90	44.27122	18.67813	38.47231	33.37398	62.93186
100	33.4231	10.19944	29.14647	25.4703	51.48153
110	25.27449	3.26572	21.07597	17.65032	42.37629
120	15.62627	0	14.95574	10.339423	33.52178
130	6.5985	0	9.2884	5.91753	23.42954
140	0	0	3.08835	1.90682	16.44772

Table D.8: Run time in seconds (number of plans visited) for different base data sets with interdicting leadership with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
50	0.671 (1)	0.656 (1)	4.188 (2)	6.672 (3)	-
60	4.766 (2)	1.437 (1)	5.468 (2)	10.359 (3)	3.813 (2)
70	1.84 (1)	36.157 (3)	4.438 (2)	29.563 (4)	4.234 (2)
80	17.64 (2)	2.031 (1)	17.062 (2)	71.203 (3)	36.266 (4)
90	54.625 (2)	2.218 (1)	176.64 (3)	60.469 (3)	25.796 (2)
100	59.375 (2)	39.797 (2)	34.672 (2)	706.828 (5)	102.11 (3)
110	79.641 (2)	21.5 (2)	167.219 (2)	555.609 (5)	47.375 (2)
120	150.265 (2)	3.031 (2)	119.984 (2)	273.141 (4)	123.75 (2)
130	163.765 (2)	0.203 (1)	153.328 (2)	350.984 (4)	206.25 (2)
140	11.094 (3)	0.141 (1)	148.078 (3)	722.672 (7)	111.968 (2)

Table D.9: Optimal flows for different data sets with organizational restructuring and interdicting leadership with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
Base	116.77137	111.89403	114.435134	110.546185	118.280546
50	98.795814	69.30883	85.118192	84.03023	-
60	86.14281	60.12165	74.435869	71.85268	101.51818
70	77.2257	49.64045	65.118192	60.58043	91.52576
80	64.723070	40.59544	54.435869	51.50233	78.08339
90	55.441619	32.54936	44.666794	38.18203	67.50571
100	46.65828	21.85086	36,691006	29.37457	57.97674
110	36.38575	13.76413	29.14647	20.19363	47.28458
120	27.08376	6.33368	21.07597	14.29806	38.26267
130	19.46085	0	14.95574	7.04445	30.4055
140	12.79303	0	8.972234	3.27475	22.50035

Table D.10: Run time in seconds (number of plans visited) for different data sets with organizational restructuring and interdicting leadership with 200 users with fixed defender budget 6

Budget	1	2	3	4	5
50	74.656 (5)	1.156 (1)	4.406 (2)	26.25 (5)	-
60	17.782 (3)	1.984 (1)	50.344 (6)	67.969 (5)	4.812 (2)
70	375.093 (6)	44.64045 (4)	68.75 (6)	61.969 (5)	606.36 (6)
80	139.875 (5)	29.015 (2)	21.406 (3)	154.484 (6)	285.156 (6)
90	102.719 (3)	783.141 (6)	52.782 (3)	244.438 (7)	412.516 (6)
100	1357.375 (6)	573.609 (5)	559.437 (5)	514.016 (5)	1192.407 (6)
110	840.975 (5)	824.36 (6)	1401.797 (7)	507 (5)	885.39 (5)
120	1600.578 (6)	926.594 (6)	625.078 (5)	2963.718 (8)	570.11 (4)
130	1661.359 (6)	9.328 (3)	1954.64 (7)	925.922 (5)	3720.812 (7)
140	2580.844 (7)	0.125 (1)	2745.125 (9)	1535.969 (9)	2004.015 (5)

APPENDIX E
RECOMMENDED INTERDICTION PLANS ON GENERATED NETWORKS

Table E.1: Interdiction recommendations from MFNIP and MFNIP-R for network 2

Attacker Budget	MFNIP Int Trafficker	MFNIP Int Bottom	MFNIP Int Victim	MFNIP-R Int Trafficker	MFNIP-R Int Bottom	MFNIP-R Int Victim
8	0	2	0	0	0	4
12	0	3	0	1	0	4
16	0	2	2	1	0	6
20	1	2	2	1	1	6
24	2	3	1	1	1	8
28	2	4	1	2	0	10
32	3	4	0	2	0	13
36	3	4	2	3	1	9
40	3	4	5	3	2	9

Table E.2: Interdiction recommendations from MFNIP and MFNIP-R for network 3

Attacker Budget	MFNIP Int Trafficker	MFNIP Int Bottom	MFNIP Int Victim	MFNIP-R Int Trafficker	MFNIP-R Int Bottom	MFNIP-R Int Victim
8	0	2	0	0	1	2
12	1	1	1	1	0	4
16	1	2	1	1	0	6
20	2	2	1	2	1	3
24	2	3	1	2	1	6
28	2	2	5	3	3	1
32	2	3	5	2	2	8
36	4	4	0	3	3	5
40	4	4	2	4	3	5

Table E.3: Interdiction recommendations from MFNIP and MFNIP-R for network 4

Attacker Budget	MFNIP Int Trafficker	MFNIP Int Bottom	MFNIP Int Victim	MFNIP-R Int Trafficker	MFNIP-R Int Bottom	MFNIP-R Int Victim
8	0	2	0	0	1	2
12	1	1	1	1	0	4
16	1	2	1	1	0	6
20	2	2	1	1	0	8
24	2	3	1	2	1	5
28	2	2	5	2	1	7
32	3	3	2	3	2	5
36	3	4	2	3	3	5
40	3	4	5	4	3	5

Table E.4: Interdiction recommendations from MFNIP and MFNIP-R for network 5

Attacker Budget	MFNIP Int Trafficker	MFNIP Int Bottom	MFNIP Int Victim	MFNIP-R Int Trafficker	MFNIP-R Int Bottom	MFNIP-R Int Victim
8	0	2	0	0	0	4
12	0	2	2	0	0	6
16	1	2	2	0	0	8
20	2	2	1	0	0	10
24	2	2	3	1	0	10
28	2	3	3	1	1	10
32	2	3	5	1	0	13
36	2	4	5	2	1	12
40	2	4	7	2	1	13

APPENDIX F
MP-MFNIP-R FULL MODEL DERIVATION

As per standard column-and-constraint generation, we separate the maximization problem into two problems, where the innermost problem in only the continuous variables, the x variables. This allows us to replace the problem with the dual minimization problem:

$$\min_{y,\gamma\in Y}\max_{z,\zeta}\min_{\pi,\theta}\sum_{t=1}^{\tau}[\sum_{i\in N\setminus\{\alpha,\omega\}}u_i\gamma_{it}\theta_{it}+\sum_{(i,j)\in A}u_{ij}\theta_{ijt}+\sum_{(i,j)\in A^{R,out}}u_{ij}\zeta^{out}_{ijt}\theta_{ijt}+\sum_{(i,j)\in A^{R,in}}u_{ij}\zeta^{in}_{ijt}\theta_{ijt}+\sum_{(i,j)\in B^R}\tilde{u_j}\zeta^{out}_{\alpha jt}\theta_{\alpha jt}] \tag{F.1a}$$

$$\text{s.t. } \pi^+_{jt}+\theta_{\alpha jt}\geq 1 \text{ for } (\alpha,j)\in A\cup A^R, t=1,\ldots,\tau \tag{F.1b}$$

$$\pi^+_{jt}-\pi^-_{it}+\theta_{ijt}\geq 0 \text{ for } (i,j)\in A\cup A^R \text{ s.t. } i\neq\alpha, j\neq\omega, t=1,\ldots,\tau \tag{F.1c}$$

$$\pi^-_{it}-\pi^+_{it}+\theta_{it}\geq 0 \text{ for } i\in N\setminus\{\alpha,\omega\}, t=1,\ldots,\tau \tag{F.1d}$$

$$-\pi^-_{it}+\theta_{i\omega t}\geq 0 \text{ for } (i,\omega)\in A\cup A^R, t=1,\ldots,\tau \tag{F.1e}$$

$$\theta\geq 0 \tag{F.1f}$$

$$y,\in\{0,1\}^{|N|\setminus\{\alpha,\omega\}} \tag{F.1g}$$

$$\gamma\in\{0,1\}^{|N|\setminus\{\alpha,\omega\}\times\tau} \tag{F.1h}$$

$$\text{Constraints (5.31i)} - \text{(5.31m)} \tag{F.1i}$$

As in Kosmas et al. [67], we use an equivalent formulation to reduce the number of bilinear terms in the objective function.

$$\min_{y,\gamma}\max_{z,\zeta}\min_{\pi,\theta}\sum_{t=1}^{\tau}[\sum_{i\in N\setminus\{\alpha,\omega\}}u_i\theta_{it}+\sum_{(i,j)\in A\cup A^{R,out}\cup A^{R,in}}u_{ij}\theta_{ijt}+\sum_{(i,j)\in B^R}\tilde{u_j}\zeta^{out}_{\alpha jt}\theta_{\alpha jt}] \tag{F.2a}$$

$$\text{s.t. } \pi^+_{jt}+\theta_{\alpha jt}\geq 1 \text{ for } (\alpha,j)\in A, t=1,\ldots,\tau \tag{F.2b}$$

$$\pi^+_{jt}-\pi^-_{it}+\theta_{ijt}\geq 0 \text{ for } (i,j)\in A \text{ s.t. } i\neq\alpha, j\neq\omega, t=1,\ldots,\tau \tag{F.2c}$$

$$\pi^-_{it}-\pi^+_{it}+\theta_{it}\geq -\gamma_{it} \text{ for } i\in N\setminus\{\alpha,\omega\}, t=1,\ldots,\tau \tag{F.2d}$$

$$-\pi^-_{it}+\theta_{i\omega t}\geq 0 \text{ for } (i,\omega)\in A, t=1,\ldots,\tau \tag{F.2e}$$

$$\pi^+_{jt}+\theta_{\alpha jt}\geq \zeta^{out}_{\alpha jt} \text{ for } (\alpha,j)\in A^{R,out}, t=1,\ldots,\tau \tag{F.2f}$$

$$\pi^+_{jt}-\pi^-_{it}+\theta_{ijt}\geq \zeta^{out}_{ijt}-1 \text{ for } (i,j)\in A^{R,out} \text{ s.t. } i\neq\alpha, j\neq\omega, t=1,\ldots,\tau \tag{F.2g}$$

$$\pi^+_{jt}-\pi^-_{it}+\theta_{ijt}\geq \zeta^{in}_{ijt}-1 \text{ for } (i,j)\in A^{R,in} \text{ s.t. } i\neq\alpha, j\neq\omega, t=1,\ldots,\tau \tag{F.2h}$$

$$\theta\geq 0 \tag{F.2i}$$

$$y,\in\{0,1\}^{|N|\setminus\{\alpha,\omega\}} \tag{F.2j}$$

$$\gamma\in\{0,1\}^{|N|\setminus\{\alpha,\omega\}\times\tau} \tag{F.2k}$$

$$\text{Constraints (5.31i)} - \text{(5.31m)} \tag{F.2l}$$

Note that $Z(y)$ is a finite set, so we can reduce the tri-level problem into a single-minimization problem by enumerating over every feasible (z, ζ) solution, and enforcing that the objective value of the bilevel problem is at least as large as objective value associated with each (z, ζ) solution. Let $M_y = |Z(y)|$. The following is a non-standard formulation of the single-level minimization problem.

$$\min_{y,\gamma,\pi,\theta} \eta \tag{F.3a}$$

$$\text{s.t. } \eta \geq \sum_{t=1}^{\tau}[\sum_{i\in N\setminus\{\alpha,\omega\}} u_i\theta_{it}^k + \sum_{(i,j)\in A\cup A^{R,out}\cup A^{R,in}} u_{ij}\theta_{ijt}^k + \sum_{(i,j)\in B^R} \tilde{u_j}\zeta_{\alpha jt}^{out,k}\theta_{\alpha jt}] \text{ for } k = 1,\ldots,M_y \tag{F.3b}$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq 1 \text{ for } (\alpha, j) \in A, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3c}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq 0 \text{ for } (i,j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3d}$$

$$\pi_{it}^{-^k} - \pi_{it}^{+^k} + \theta_{it}^k \geq -\gamma_{it} \text{ for } i \in N \setminus \{\alpha,\omega\}, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3e}$$

$$-\pi_{it}^{-^k} + \theta_{i\omega t}^k \geq 0 \text{ for } (i,\omega) \in A, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3f}$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq \zeta_{\alpha jt}^{k,out} \text{ for } (\alpha, j) \in A^{R,out}, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3g}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq \zeta_{ijt}^{k,out} - 1 \text{ for } (i,j) \in A^{R,out} \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3h}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq \zeta_{ijt}^{k,in} - 1 \text{ for } (i,j) \in A^{R,in} \text{ s.t. } i \neq \alpha, j \neq \omega, k = 1,\ldots,M_y, t = 1,\ldots,\tau \tag{F.3i}$$

$$\theta \geq 0 \tag{F.3j}$$

$$y, \in \{0,1\}^{|N|\setminus\{\alpha,\omega\}} \tag{F.3k}$$

$$\gamma \in \{0,1\}^{|N|\setminus\{\alpha,\omega\}\times\tau} \tag{F.3l}$$

$$\text{Constraints (5.31i)} - \text{(5.31m)} \tag{F.3m}$$

To return to a standard formulation, we now include our partial information constraints and variables. The include of these constraints and variables enforce that, for a given y and $(z, \zeta) \notin Z(y)$, we identify another solution $(\bar{z}, \bar{\zeta}) \in Z(y)$ that allows for nonzero components of (z, ζ) that are feasible with respect to y will remain nonzero. These constraints are (5.32k)-(5.32t). Let $M = \bigcup_{y\in Y} Z(y)$. The following is a standard formulation of the single-level problem with partial information constraints.

$$\min_{y,\gamma,w\pi,\theta} \eta \tag{F.4a}$$

$$\text{s.t. } \eta \geq \sum_{t=1}^{\tau}[\sum_{i\in N\setminus\{\alpha,\omega\}} u_i\theta_{it}^k + \sum_{(i,j)\in A\cup A^{R,out}\cup A^{R,in}} u_{ij}\theta_{ijt}^k + \sum_{(i,j)\in B^R} \tilde{u_j} z_{\alpha jt}^{out,k} w_{\alpha jt}^{out,k}\theta_{\alpha jt}^k] \tag{F.4b}$$

$$\text{for } k = 1,\ldots,M$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq 1 \text{ for } (\alpha, j) \in A, k = 1,\ldots,M, t = 1,\ldots,\tau \tag{F.4c}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq 0 \text{ for } k = 1,\ldots,M, (i,j) \in A \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1,\ldots,\tau \tag{F.4d}$$

$$\pi_{it}^{-^k} - \pi_{it}^{+^k} + \theta_{it}^k \geq -\gamma_{it} \text{ for } i \in N \setminus \{\alpha, \omega\}, k = 1, \ldots, M, t = 1, \ldots, \tau \tag{F.4e}$$

$$-\pi_{it}^{-^k} + \theta_{i\omega t}^k \geq 0 \text{ for } (i, \omega) \in A, k = 1, \ldots, M, t = 1, \ldots, \tau \tag{F.4f}$$

$$\pi_{jt}^{+^k} + \theta_{\alpha jt}^k \geq w_{\alpha jt}^{out,k} + z_{\alpha j}^{out,k} - 1 \text{ for } (\alpha, j) \in A^{R,out}, k = 1, \ldots, M, t = 1, \ldots, \tau \tag{F.4g}$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq w_{ijt}^{out,k} + z_{ij}^{out,k} - 2 \tag{F.4h}$$

$$\text{for } k = 1, \ldots, M, (i, j) \in A^{R,out} \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau$$

$$\pi_{jt}^{+^k} - \pi_{it}^{-^k} + \theta_{ijt}^k \geq w_{ijt}^{in,k} + z_{ij}^{in,k} - 2 \tag{F.4i}$$

$$\text{for } k = 1, \ldots, M, (i, j) \in A^{R,in} \text{ s.t. } i \neq \alpha, j \neq \omega, t = 1, \ldots, \tau$$

$$\theta^k \geq 0 \text{ for } k = 1, \ldots, M \tag{F.4j}$$

$$\mu_i^{out} \lambda_{it}^{out,k} + \sum_{(i,j) \in A^{R,out}: z_{ij}^{k,out} = 1} w_{ij(t+\delta_{ij}^{z,out})}^{out,k} \geq \sum_{(i,h) \in A} \gamma_{ht} \tag{F.4k}$$

$$\text{for } k \in 1, \ldots, M, i \in T, t \in 1, \ldots, \tau - \delta_i^{max,out}$$

$$\mu_j^{in} \lambda_{jt}^{in,k} + \sum_{(i,j) \in A^{R,in}: z_{ij}^{k,in} = 1} w_{ij(t+\delta_{ij}^{z,in})}^{in,k} \geq \sum_{(h,j) \in A} \gamma_{ht} \tag{F.4l}$$

$$\text{for } k \in 1, \ldots, M, j \in V, t \in 1, \ldots, \tau - \delta_i^{max,in}$$

$$\lambda_{it}^{out,k} \leq \frac{\sum_{(i,j) \in A^{R,out}: z_{ij}^{k,out} = 1} w_{ij(t+\delta_{ij}^{z,out})}^{out,k}}{\sum_{(i,j) \in A^{R,out}: z_{ij}^{k,out} = 1} \bar{z}_{ij}^{out,k}} \text{ for } k \in 1, \ldots, M, i \in T, t = 1, \ldots, \tau \tag{F.4m}$$

$$\lambda_{it}^{in,k} \leq \frac{\sum_{(i,j) \in A^{R,in}: z_{ij}^{k,in} = 1} w_{ij(t+\delta_{ij}^{z,in})}^{in,k}}{\sum_{(i,j) \in A^{R,in}: z_{ij}^{k,in} = 1} \bar{z}_{ij}^{in,k}} \text{ for } k = 1, \ldots, M, i \in V, t = 1, \ldots, \tau \tag{F.4n}$$

$$w_{ij}^{out,k} \geq w_{ij(t-1)}^{out,k} \text{ for } k = 1, \ldots, M, (i, j) \in A^{R,out}, t \in 2 \ldots, \tau \tag{F.4o}$$

$$w_{ijt}^{in,k} \geq w_{ij(t-1)}^{in,k} \text{ for } k = 1, \ldots, M, (i, j) \in A^{R,in}, t \in 2 \ldots, \tau \tag{F.4p}$$

$$w_{\alpha jt}^{out,k} \geq y_i \text{ for } k = 1, \ldots, M, (j, i) \in T^R, t \in \delta_i^y + \delta_{\alpha j}^{z,out}, \ldots, \tau \tag{F.4q}$$

$$w_{\alpha j(t+\delta_{\alpha j}^{z,out})}^{out,k} \geq y_i \text{ for } k = 1, \ldots, M, (j, i) \in B^R \text{ s.t. } z_{\alpha j}^{out,k} = 1, t \in \delta_i^y + 1, \ldots, \tau - \delta_{\alpha j}^{z,out} \tag{F.4r}$$

$$w_{ijt}^{out,k} \geq \bar{z}_{kij}^{out,k} \text{ for } k = 1, \ldots, M, i \in B, j \in V \text{ s.t. } (i, j) \in A^{R,out} \tag{F.4s}$$

$$w_{ijt}^{out,k} \geq \bar{z}_{ij}^{out,k} \text{ for } k = 1, \ldots, M, i \in V, j \in V \text{ s.t. } (i, j) \in A^{R,out}, \exists l \in B, (i, l) \in B^R \tag{F.4t}$$

$$y, \in \{0, 1\}^{|N| \setminus \{\alpha, \omega\}} \tag{F.4u}$$

$$\gamma \in \{0, 1\}^{|N| \setminus \{\alpha, \omega\} \times \tau} \tag{F.4v}$$

$$\text{Constraints (5.31i)} - \text{(5.31m)} \tag{F.4w}$$

In general, M will be very large, making the problem computationally difficult to solve. We can instead optimize over a subset of points $\{z^1, \zeta^1, \ldots, z^n, \zeta^n\} \subset \bigcup_{y \in Y} Z(y)$, where $n < M$ to identify a lower bound on the true objective value of the bilevel problem. We then iteratively identify new points $(\tilde{z}, \tilde{\zeta})$ to include in the set we optimize over, until the true solution is identified. This is the minimization problem outlined in Section 5.5.

APPENDIX G
MP-MFNIP-R FULL COMPUTATIONAL RESULTS

Table G.1: Recommended delayed interdictions on network 1 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	1	2	0	1	2
12	0	1	4	0	2	2
16	0	2	4	1	0	6
20	0	1	8	1	0	8
24	0	1	10	1	0	10
28	0	2	10	1	1	10
32	0	1	14	0	0	16
36	0	2	14	0	0	18
40	0	2	16	0	0	20

Table G.2: Recommended delayed interdictions on network 2 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	1	2
12	0	2	2	0	2	2
16	0	3	2	1	0	6
20	0	2	6	1	0	8
24	0	2	8	1	0	10
28	0	2	10	2	0	10
32	0	2	12	2	0	12
36	0	3	12	0	0	18
40	1	2	14	1	0	18

Table G.3: Recommended delayed interdictions on network 3 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	2	0
12	0	3	0	0	0	6
16	0	2	4	1	0	6
20	0	2	6	1	0	8
24	1	2	6	0	0	12
28	2	2	5	1	0	12
32	2	2	7	1	0	14
36	2	2	9	1	0	16
40	1	2	14	1	0	18

Table G.4: Recommended delayed interdictions on network 4 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	2	0
12	1	1	2	0	0	6
16	1	1	4	1	0	6
20	2	2	1	1	0	8
24	2	3	1	1	0	10
28	1	1	10	2	1	8
32	2	3	5	2	1	10
36	2	2	9	2	0	14
40	2	3	9	2	0	16

Table G.5: Recommended delayed interdictions on network 5 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	1	2	0	0	4
12	0	1	4	0	0	6
16	0	1	6	0	0	8
20	0	1	8	0	0	10
24	0	1	10	1	0	10
28	0	2	10	1	1	10
32	0	1	14	0	0	16
36	0	2	14	0	0	18
40	0	3	14	1	1	16

Table G.6: Recommended upfront interdictions on network 1 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	0	4
12	1	1	1	1	0	4
16	1	2	1	1	0	6
20	1	3	1	1	0	8
24	0	2	8	1	0	10
28	0	2	10	1	1	10
32	0	2	12	2	2	8
36	1	2	10	2	2	10
40	1	2	12	2	3	10

Table G.7: Recommended upfront interdictions on network 2 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	0	4
12	0	3	0	1	0	4
16	0	4	0	1	0	6
20	0	4	2	1	1	6
24	1	4	1	1	0	10
28	2	4	1	2	0	10
32	2	4	3	2	0	12
36	0	3	12	2	1	12
40	0	4	12	2	1	14

Table G.8: Recommended upfront interdictions on network 3 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	0	4
12	0	3	0	1	1	2
16	0	2	4	1	0	6
20	1	2	3	1	0	8
24	1	2	5	1	0	10
28	2	2	5	1	0	12
32	2	3	5	1	0	14
36	2	2	9	1	0	16
40	2	3	9	1	0	18

Table G.9: Recommended upfront interdictions on network 4 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	2	0	0	0	4
12	1	1	2	1	0	4
16	1	2	1	1	0	6
20	2	2	1	1	0	8
24	2	3	1	1	1	8
28	2	3	3	2	1	8
32	2	3	5	2	1	10
36	2	3	7	2	0	14
40	2	3	9	2	1	14

Table G.10: Recommended upfront interdictions on network 5 with one attacker

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim
8	0	1	2	0	0	4
12	0	1	4	0	0	6
16	0	1	6	0	0	8
20	0	1	8	0	0	10
24	0	2	8	1	0	10
28	0	3	8	1	1	10
32	0	4	8	0	2	12
36	0	3	12	2	1	12
40	0	3	14	2	1	14

Table G.11: Recommended delayed interdictions on network 1 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	1	2	3	1	0	0	4	2	3
12	0	1	4	2	3	0	0	6	2	3
16	0	1	6	3	1	0	0	8	2	3
20	0	1	8	3	1	0	0	10	2	3
24	0	2	12	2	3	0	0	12	2	4
28	0	1	14	2	3	0	0	14	2	4
32	0	1	14	2	4	0	0	16	2	4
36	0	2	14	2	4	0	0	18	2	4
40	0	3	14	2	4	0	0	20	0	8

Table G.12: Recommended delayed interdictions on network 2 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	4
12	0	2	2	3	1	0	0	6	2	4
16	0	2	4	3	1	0	0	8	2	4
20	0	2	6	2	3	0	0	10	1	6
24	0	2	8	2	4	0	0	12	2	4
28	0	2	10	2	4	0	0	14	1	7
32	0	3	10	2	4	0	0	16	1	6
36	1	2	12	2	4	0	0	18	1	6
40	0	1	18	2	4	0	0	20	1	7

Table G.13: Recommended delayed interdictions on network 3 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	3
12	0	2	2	3	1	0	0	6	2	3
16	0	2	4	3	1	0	0	8	2	4
20	0	2	6	3	1	0	0	10	2	4
24	1	2	6	3	1	0	0	12	2	4
28	0	1	12	2	4	0	0	14	2	4
32	1	3	8	2	4	0	0	16	2	4
36	2	3	7	2	4	0	0	18	1	7
40	1	1	14	2	4	0	0	20	1	7

Table G.14: Recommended delayed interdictions on network 4 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	1	2	3	1	0	0	4	2	3
12	1	1	2	3	1	0	0	6	2	4
16	0	1	6	2	3	0	0	8	2	4
20	2	2	1	3	1	0	0	10	2	4
24	2	2	3	3	0	0	0	12	1	7
28	1	2	8	2	4	1	0	12	2	3
32	1	2	10	2	4	1	0	14	2	4
36	0	0	18	1	7	1	0	16	2	4
40	0	1	18	1	7	1	0	18	2	4

Table G.15: Recommended delayed interdictions on network 5 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	1	2	3	1	0	0	4	3	1
12	0	1	4	3	1	0	0	6	3	1
16	0	1	6	3	1	0	0	8	2	3
20	0	1	8	3	1	0	0	10	2	3
24	0	1	10	2	4	0	0	12	2	4
28	0	1	12	2	4	0	0	14	1	7
32	0	1	14	1	7	0	0	16	1	7
36	0	2	14	1	7	0	0	17	1	7
40	0	3	14	1	7	0	1	17	1	7

Table G.16: Recommended upfront interdictions on network 1 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	3
12	0	2	2	2	2	0	0	6	2	3
16	0	2	4	2	3	0	0	8	2	3
20	0	2	6	2	3	0	0	10	2	3
24	0	2	8	2	3	0	0	12	2	4
28	0	2	10	2	3	0	0	14	2	4
32	0	3	10	2	3	0	0	16	2	4
36	0	2	14	2	4	0	0	18	2	4
40	0	3	14	2	4	0	0	20	0	8

Table G.17: Recommended upfront interdictions on network 2 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	4
12	0	2	2	3	1	0	0	6	2	4
16	0	4	0	3	1	0	0	8	2	4
20	0	3	4	2	3	0	0	10	1	5
24	0	3	6	2	4	0	0	12	1	7
28	0	3	8	2	4	0	0	14	1	6
32	0	3	10	2	4	0	0	16	1	7
36	0	3	12	1	5	0	0	18	1	6
40	1	3	12	2	4	0	0	20	1	7

Table G.18: Recommended upfront interdictions on network 3 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	3
12	0	2	2	3	1	0	0	6	2	3
16	1	2	1	3	1	0	0	8	2	4
20	2	2	1	3	1	0	0	10	2	4
24	1	2	5	3	1	0	0	12	2	4
28	2	2	5	3	1	0	0	14	2	4
32	1	3	7	2	4	0	0	16	2	4
36	2	3	7	2	4	0	0	18	1	7
40	2	3	9	1	5	0	0	20	1	7

Table G.19: Recommended upfront interdictions on network 4 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	2	0	3	1	0	0	4	2	4
12	1	1	1	3	1	0	0	6	2	4
16	1	2	2	3	1	0	0	8	2	4
20	2	2	1	3	1	0	0	10	2	4
24	2	3	1	2	2	1	0	10	2	3
28	2	3	3	2	3	1	0	12	2	3
32	2	3	5	2	4	1	0	14	2	4
36	2	3	7	2	4	1	0	16	2	4
40	2	3	9	2	4	0	0	20	0	10

Table G.20: Recommended upfront interdictions on network 5 with two attackers

Budget	MMFNIP Int Trafficker	MMFNIP Int Bottom	MMFNIP Int Victim	MMFNIP 2nd Int Current Vic	MMFNIP 2nd Int Prosp Vic	MP-MFNIP-R Int Trafficker	MP-MFNIP-R Int Bottom	MP-MFNIP-R Int Victim	MP-MFNIP-R 2nd Int Current Victim	MP-MFNIP-R 2nd Int Prosp Victim
8	0	1	2	3	1	0	0	4	3	1
12	0	1	4	3	1	0	0	6	3	1
16	0	2	4	3	1	0	0	8	2	3
20	0	2	6	2	2	0	0	10	2	4
24	0	3	6	2	3	0	0	12	2	4
28	0	3	8	2	4	0	0	14	1	5
32	0	3	10	1	7	0	0	16	1	7
36	0	3	12	1	7	0	1	16	1	7
40	0	4	12	1	7	0	2	16	1	7

Table G.21: MP-MFNIP-R flow with delayed interdiction and one attacker

Budget	Data1	Data2	Data3	Data4	Data5
Base	161	189	210	210	154
8	122	164	185	181	130
12	116	155	177	172	118
16	105	146	167	160	114
20	98	138	155	150	102
24	86	128	149	143	94
28	81	122	139	135	89
32	77	111	129	125	82
36	70	104	120	114	74
40	62	96	109	107	69

Table G.22: MP-MFNIP-R flow with delayed interdiction and two attackers

Budget	Data1	Data2	Data3	Data4	Data5
Base	161	189	210	210	154
8	109	146	167	160	116
12	98	138	155	152	104
16	86	126	147	140	94
20	79	116	135	134	82
24	68	108*	125	129*	74
28	61	97	116	114	64
32	54	88	105	106	52
36	46	74	94	94	46
40	38	66	82	87*	41

Table G.23: MP-MFNIP-R flow with upfront interdiction and one attacker

Budget	Data1	Data2	Data3	Data4	Data5
Base	161	189	210	210	154
8	119	161	185	183	126
12	110	153	173	167	112
16	97	139	159	154	108
20	89	130	147	140	94
24	77	119	140	133	84
28	70	110	129	121	77
32	65	98	117	110	71
36	54	88	106	98	62
40	47	80	94	91	54

Table G.24: MP-MFNIP-R flow with upfront interdiction and two attackers

Budget	Data1	Data2	Data3	Data4	Data5
Base	161	189	210	210	154
8	104	140	161	154	110
12	91	131	147	145	96
16	77	117	138	131	84
20	69	105	124	124	70
24	56	96	112	112	61
28	48	83	102	98	49
32	40	70	89	89	35
36	31	56	76	75	27
40	21	47	62	68	20

APPENDIX H
SUPPLEMENTAL FILES

H.1 Permissions for Chapters 1 and 2

This file contains license details and terms and conditions for the reproduction of material used in Chapter 1 and Chapter 2 of this dissertation.

File name: Networks Permission to Republish.pdf

File type: Portable Document Format (PDF)

File size: 262 KB

Required application software: Adobe Acrobat or any standard PDF viewer

Special hardware requirements: None