Coordinating resources in Stackelberg Security Games

Víctor Bucarey L.[a,b], Carlos Casorrán[a,b], Martine Labbé[a,b], Fernando Ordoñez[c], Oscar Figueroa[d]

[a] *Département d'Informatique, Université Libre de Bruxelles, Brussels, Belgium.*
[b] *Inria Lille-Nord Europe, Villeneuve d'Ascq, France.*
[c] *Departamento de Ingeniería Industrial, Universidad de Chile, Santiago, Chile.*
[d] *Carabineros de Chile, Santiago, Chile.*

Abstract

In this work we formulate a Stackelberg Security Game that coordinates resources in a border patrol problem. In this security domain, resources from different precincts have to be paired to conduct patrols in the border due to logistic constraints. Given this structure, models that enumerate the pure defender strategies scale poorly. We describe the set of mixed strategies using a polynomial number of variables but exponentially many constraints that come from the matching polytope. We then include this description in a mixed integer formulation to compute the Strong Stackelberg Equilibrium efficiently with a branch and cut scheme. Since the optimal patrol solution is a probability distribution over the set of exponential size, we also introduce an efficient sampling method that can be used to deploy the security resources every shift. Our computational results evaluate the efficiency of the branch and cut scheme developed and the accuracy of the sampling method.

Keywords: OR in Defence, Bilevel Optimization, Stackelberg Games

1. Introduction

In this article we consider the operational problem of patrolling the border in the presence of strategic adversaries that seek to trespass it taking into account the defender patrols. We model the situation as a Stackelberg Security Game (SSG) where the defender acts as the leader of a Stackelberg Game (SG), executing a preventive border patrol, which is observed prior to the optimal response by the strategic adversary, which acts as the follower. Due to the size of the border, the defender coordinates local resources to achieve a global defender strategy. Motivated by a real border patrol problem, we consider that resources from adjacent precincts are paired together to patrol a location in their area.

URL: vbucarey@ulb.ac.be (Víctor Bucarey L.), ccasorra@ulb.ac.be (Carlos Casorrán), mlabbe@ulb.ac.be (Martine Labbé), fordon@dii.uchile.cl (Fernando Ordoñez), oscar.figueroa@carabineros.cl (Oscar Figueroa)

1.1. Motivational Example

In order to motivate the problem, we describe a realistic border patrol problem proposed by Carabineros de Chile, which is the Chilean national police force. In this problem, Carabineros considers three different types of crime: drug trafficking, contraband of goods or animals and the illegal entry of people. Among other activities, Carabineros conducts patrols along the border with the aim of preventing the occurrence of these border crimes.

Here we describe the specific planning of night patrols. Carabineros has identified a set of locations along the border that can serve as vantage points from where to conduct surveillance with technical equipment such as night goggles and heat sensors. A night shift action consists of deploying personnel to these locations. Due to the large distances and harsh landscape, the patrol detail is stationed at these locations overnight, forcing a commitment of these resources for a long period. The border region is divided into several police precincts. Due to the vast expanses and the lack of manpower, for the purpose of conducting night patrols, the border precincts are paired up, so each precinct that commits resources is not overburdened by this preventive patrolling action. A joint patrol detail, combining personnel from paired precincts, conducts surveillance at one of the vantage points within the paired precincts territory. These border patrols are planned weekly.

We describe an instance representing the border patrol problem for the Arica Parinacota Region of Chile. This region, which is Chile's northernmost region, is depicted in Figure 1. This region has international borders both with Peru and Bolivia. The border begins at the Pacific Ocean and goes to the north-east separating Chile from Peru while climbing to more than 4000 meters above the sea level. It then goes north-south in the Andean Plateau establishing the limit between Chile and Bolivia.

In Figure 1 left, we illustrate a realistic instance where possible border precincts are enumerated from 1 to 8. In the right side, we give the graph representation that identifies possible precinct pairings. In this graph the possible outposts in each precinct are denoted by letters `[a]` to `[u]`. For instance, if Carabineros decide at the tactical level to camp every week at three points, then a possible operational plan for this week can be to pair precincts 1 and 2 to camp in `[c]`, pair precincts 4 and 5 to protect `[h]` and to pair 7 and 8 to protect `[u]`. The set of all possible actions that Carabineros can perform is given by all the possible pairings of size $m = 3$, and the possible $m = 3$ outpost that are covered by the pairings. In this instance this set has 901 elements. Due to national security reasons we do not expose the real configuration of precincts and outposts.

Carabineros estimate the frequency in which they face each of the three types of crimes. This is used to estimate the probability of facing each type of attacker. To evaluate the police strategy, the benefits and losses of the interaction with each attacker type are also estimated. This information is summarized in payoff matrices which are built using historic information of crime occurrences detailing

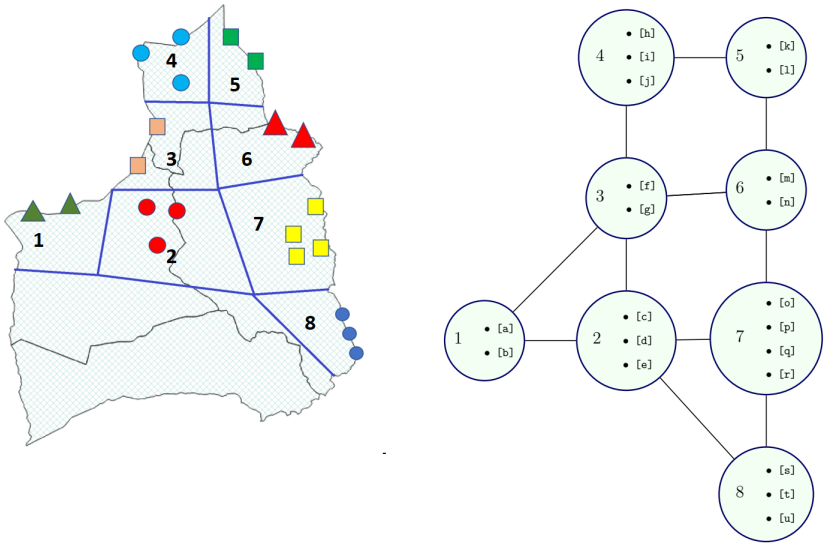


Figure 1: Instance in the context of border patrolling and its graph representation.

location and a description of the crime, the patrol locations, and information on border foot paths and trails provided by police. Open source information on population estimates in Chile and across the border, police precinct boundaries, commercial valuations of goods and punishment for crimes in Chile is also taken into account ([7, 1]). A detailed description of how this instance is constructed appears in [4].

1.2. Problem statement

The problem we tackle in this paper corresponds to the setting where a defender deploys limited resources to protect targets that can be attacked by an adversary, motivated by the application described above. We model this situation as a Stackelberg Game where the leader plays the role of the defender and the follower represents the attacker (which can be from one of many types corresponding to different types of crime). We consider a border security problem where a patrolling team is formed from resources originating from adjacent police precincts. The combined resource is deployed to patrol a target inside one of the two precincts.

To represent the coordination in our border patrol problem, we consider a graph composed of nodes for the police precincts and edges that connect two precincts if they can be paired. Within each precinct there is a set of locations, or targets, that can be patrolled. In this case the set of defender pure strategies corresponds to the set of all matchings of size m together with the corresponding possible targets that are protected. The defender aims to select the mixed strategy over this set of actions that optimizes its payoff taking into account that the adversary will take this solution into account when deciding its own best action.

The appropriate solution concept for this problem is the Strong Stackelberg Equilibria (SSE) of the game, that is, strategies for the leader and each attacker type that satisfy: i. The leader plays a payoff-maximizing strategy; ii. Each follower type plays a best-response strategy; and iii. Each follower, when indifferent between best-response strategies, selects the one that maximizes the objective of the leader. An SSE solution always exists in our setting, since it is the solution of a bounded optimization problem, see Section 2 below.

1.3. Literature Review

According to [9], securing national borders to defend from the illegal movement of contraband, drugs and people is a natural concern of nations. The European Union (EU) created the European Border and Coast Guard in October 2016 in response to increases in migrant flows into the EU. The European Border and Coast Guard lists as one of its prime objectives "organizing joint operations and rapid border interventions to strengthen the capacity of the member states to control the external borders, and to tackle challenges at the external border resulting from illegal immigration or cross-border crime". Such operations involve different resources possibly from different locations in a joint effort to coordinate a global border plan. It is crucial to balance the effectiveness of a global plan with the cost and difficulty of locally coordinating resources in undermanned areas. As is stated in [11], the United States the Department of Homeland Security states as a primary objective "protecting [the] borders from the illegal movement of weapons, drugs, contraband, and people, while promoting lawful entry and exit" claiming it is "essential to homeland security, economic prosperity, and national sovereignty".

Authors in [28] give a survey of optimization and mathematical models that have been used in diverse homeland security applications to efficiently assign limited security resources to defend systems and infrastructure from attack. Previous work that addresses operational decisions in homeland security applications include [13] and [30] that consider optimization models for container screening and port security respectively. Passenger screening decisions in airports is addressed in [22] by an integer optimization model that uses risk level grouping and in [21] where adaptive risk-based screening policies are determined via Dynamic Programming.

Models that explicitly take into account the strategic nature of the adversary lead to game theory models, which include network interdiction problems, [12], and patrolling games over networks, [2]. In particular, in [24], the authors determine the optimal patroling strategy on a border by considering a zero sum simultaneous game over a line network. Another area of research that is related to border patrols are inspection and smuggler games [27, 5]. In this work multi-stage games are used to model the interaction between smugglers and officers, and closed form solutions are found for the zero-sum simultaneous game equilibrium.

Stackelberg Games are an example of bilevel optimization programs [3]. In these games, the top level decisions are made by a player, named the leader, that takes into account the optimal solution performed by a player named follower. Recent research has used SG to model and provide implementable defender strategies in real life security applications. In these applications, a defender aims to defend targets from a strategic adversary by deploying limited resources to protect them. Examples of SSG applications include assigning Federal Air Marshals to transatlantic flights, [18], determining U.S. Coast Guard patrols of port infrastructure [26], police patrols to prevent fare evasion in public transport systems [33], as well as protecting endangered wildlife [31].

One of the challenges that has to be addressed in solving these SG for real-world security applications is problem size. SG and SSG are known to be NP-hard [8, 20]. In [29], a relevant discussion is introduced linking the structure of the set of pure strategies of the defender and the complexity of computing the SSE.

Even in the simplest case, where the defender action is to allocate resources to targets, to enumerate all the possible actions can be untractable. In [19] a relaxation of the Stackelberg security game is formulated which determines the frequency with which each target is protected. Decomposition approaches have been developed for the general case, when feasible defender actions satisfy additional constraints. For instance, a column generation approach is introduced in [17] and a cutting plane approach in [32]. A branch and cut approach based on Benders' decomposition is introduced in [34]. A specialized decomposition method that exploits specific problem structure was introduced in [16].

1.4. Contributions

The contributions of this work include the introduction of a new mixed integer optimization formulation for SSG that allow coordinate defender strategies. We provide a discussion of the complexity of this problem. To address poor scaling of the models we introduce equivalent compact formulations that have a polynomial number of variables but exponentially many matching constraints. Furthermore, to efficiently solve these compact formulations, we adapt the [23] branch and cut scheme. To execute the optimal defender strategies, it is necessary to sample from the exponential size defender strategy space. We present both an exact, column generation based, approach to sample from the defender strategy space and an approximate method. Our computational results explore the efficiency of solving the SG for coordinated defender strategies and the accuracy of the approximate defender strategy sampling method. A preliminary work appeared in the conference [4], where a first model without any proof of correctness is provided.

1.5. Structure of the Article

The structure of the paper is as follows: In Section 2 we state the Mixed Integer formulations for this problem. Section 3 is divided in three sections: first, we described a methodology to retrieve

coverage distributions into probability distributions in the original strategy space; secondly, we outline the branch and cut algorithm used to solve these formulations; and finally, we provide some complexity results. In Section 4 we prove the correctness of formulations by showing that they are equivalent to Stackelberg security games when the defender strategies require pairing of resources. In Section 5 we discuss methods to sample from the exponential size defender strategy space. We present both an exact method and a computationally efficient approximate sampling method. Section 6 presents computational experiments to evaluate the difficulty of solving the different problem formulations and the accuracy and efficiency of the sampling methods. Finally, we present our conclusions and discuss future work in Section 7.

2. Problem Formulations

In this section we present different formulations of the SSG problem with coordinating resources. First, we present formulations for general Stackelberg games where a complete enumeration of the pure strategy set of the leader is needed. These formulations were introduced in [25, 6]. Then, we present compact formulations of SSG with coordinating resources which consider fewer variables but more constraints in order to improve the computational performance. First, we describe the model (BP), presented in [4], and then introduce a new stronger formulation (COMB).

2.1. Formulations for general SG

We consider a general Bayesian Stackelberg game, where a leader is facing a set K of followers, as introduced in [25]. In this setting, the leader commits to a payoff-maximizing strategy, anticipating that every follower knows the leader's choice and responds by selecting their own payoff-maximizing strategy. The solution concept we follow in this game is the *strong Stackelberg equilibrium*. In this equilibrium followers select a best response that favors the leader when they are indifferent between several optimal strategies.

In this model the leader knows the probability π_k of facing follower $k \in K$. We denote by I the finite set of pure strategies for the leader and by J be the finite set of pure strategies for each of the followers. A mixed strategy for the leader consists in a vector $\mathbf{x} = (x_i)_{i \in I}$, such that x_i is the probability that the leader plays pure strategy i. Analogously, a mixed strategy for follower $k \in K$ is a vector $\mathbf{q}^k = (q_j^k)_{j \in J}$ such that q_j^k is the probability that follower k plays pure strategy j. The payoffs are represented by the payoff matrices $(R^k, C^k)_{k \in K}$, with $R^k, C^k \in \mathbb{R}^{|I| \times |J|}$. The R_{ij}^k (C_{ij}^k) entry gives the reward for the leader (k-th follower) of taking the leader action i and the k-th follower action j. With these payoff matrices, given a mixed strategy $\mathbf{x}$ for the leader and strategy $\mathbf{q^k}$ for follower k, the expected utility for follower k is given by $\sum_{i \in I} \sum_{j \in J} C_{ij}^k x_i q_j^k$ while the expected utility

for the leader is given by $\sum_{k\in K}\pi_k\sum_{i\in I}\sum_{j\in J}R^k_{ij}x_iq^k_j$. It can be shown that in a SSE always exists a pure strategy that is a best response for the follower.

Authors in [25] introduce the following formulation:

$$\text{(D2)}\quad \max_{x,q,s,f}\quad \sum_{k\in K}\pi^k f^k \tag{2.1}$$

$$\text{s.t.}\quad \mathbf{x}^\top\mathbf{1}=1,\ \mathbf{x}\geq 0, \tag{2.2}$$

$$\mathbf{q}^{k\top}\mathbf{1}=1,\ \mathbf{q}^k\in\{0,1\}^{|J|} \qquad \forall k\in K \tag{2.3}$$

$$f^k\leq\sum_{i\in I}R^k_{ij}x_i+M(1-q^k_j) \qquad \forall j\in J,\ \forall k\in K, \tag{2.4}$$

$$0\leq s^k-\sum_{i\in I}C^k_{ij}x_i\leq M(1-q^k_j) \qquad \forall j\in J,\ \forall k\in K. \tag{2.5}$$

$$\mathbf{s},\mathbf{f}\in\mathbb{R}^K\ , \tag{2.6}$$

where variables f^k and s^k represent the expected utility for the leader and the follower of type k respectively when they face each other. The objective function maximises the leader's expected utility. Constraints (2.2) defines the mixed strategy of the leader. For each follower of type k, expressions (2.3) state their pure response. Constraints (2.4) upper bound the expected utility of the leader when she faces the follower type k. This bound is tight for the action j such that $q^k_j=1$. Finally, constraints (2.5) ensure that for each follower k will best respond to a strategy $\mathbf{x}$.

The (MIP-G) equivalent formulation, introduced in [6], is constructed by using the linearizing variable $h^k_{ij}=x_iq^k_j$ and dropping out variables $\mathbf{f}$ and $\mathbf{s}$ through Fourier-Motzkin Elimination. The formulation is as follows:

$$\text{(MIP-G)}\quad \max_{h,q}\quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}\pi^kR^k_{ij}h^k_{ij} \tag{2.7}$$

$$\text{s.t.}\quad \sum_{j\in J}h^k_{ij}=\sum_{j\in J}h^1_{ij} \qquad \forall i\in I,\forall k\in K \tag{2.8}$$

$$\sum_{i\in I}h^k_{ij}=q^k_j,\ \mathbf{h}\geq 0 \qquad \forall j\in J,\forall k\in K \tag{2.9}$$

$$\mathbf{q}^{k\top}\mathbf{1}=1,\ \mathbf{q}^k\in\{0,1\}^{|J|} \qquad \forall k\in K \tag{2.10}$$

$$\sum_{i\in I}\left(C^k_{ij}-C^k_{il}\right)h^k_{ij}\geq 0 \qquad \forall j,l\in J,\ \forall k\in K, \tag{2.11}$$

where the objective function (2.7) maximizes the expected defender reward. Constraints (2.8) ensures that $x_i=\sum_{j\in J}h^k_{ij}$, that is each attacker type observes the same defender strategy. Constraints (2.9) represents the relationship between variables $\mathbf{h}$ and $\mathbf{q}$. Finally, constraints (2.11) states the attacker's best response.

While (MIP-G) has more variables and constraints, [6] shows that it provides a tighter linear optimization relaxation. We point the reader to that work for a detailed study of the strength and efficiency of these formulations.

2.2. Resource Combination in a SSG

A SSG assumes two things: 1) that the leader (defender) has to protect n targets by deploying resources capable of protecting up to $m < n$ of them. The follower (attacker) selects one target to attack. 2) the rewards depend only on whether the target attacked is protected or not. In this case the set of follower strategies is $J = \{1, \ldots, n\}$ and the strategy set for the defender consists of all feasible deployments that protect up to m targets simultaneously. That is $I \subset \{i \mid i \subset J,\ |i| \leq m\}$ since there might be logistical constraints that prohibit all combinations of m targets to be protected simultaneously. Let $j \in i$ denote the condition that defender strategy i patrols target j. We express the rewards in SSG as follows:

$$R_{ij}^k = \begin{cases} D^k(j|p) & \text{if } j \in i \\ D^k(j|u) & \text{if } j \notin i \end{cases} \qquad (2.12) \qquad C_{ij}^k = \begin{cases} A^k(j|p) & \text{if } j \in i \\ A^k(j|u) & \text{if } j \notin i. \end{cases} \qquad (2.13)$$

Here $D^k(j|p)$ ($D^k(j|u)$) denotes the utility of the defender when target $j \in J$ is protected (unprotected) and is attacked by attacker $k \in K$. Similarly, the utility of an attacker of type $k \in K$ when attacking a protected (an unprotected) target $j \in J$ is denoted by $A^k(j|p)$ ($A^k(j|u)$).

We now formulate the SSG problem where the defender first teams up resources from different precincts to form m combined patrols and then decides where to deploy them. Let V be the set of police precincts. We let $E \subset V \times V$ be the set of edges representing the set of possible precinct pairings, forming an adjacency graph $G = (V, E)$. We denote by $\delta(v) \subset E$ the set of edges incident to precinct $v \in V$, similarly for any $U \subset V$, $\delta(U) \subset E$ denotes the edges between U and $V \setminus U$, and $E(U) \subset E$ denotes the edges between precincts in U. We can then represent the possible combinations of m precincts pairs as the set of matchings of size m, which is given by:

$$\mathcal{M}_m := \left\{ \mathbf{y} \in \{0,1\}^{|E|} : \sum_{e \in E} y_e = m,\ \sum_{e \in \delta(v)} y_e \leq 1 \quad \forall v \in V \right\}.$$

For every precinct $v \in V$, let J_v be the set of targets in that precinct. Note that $\{J_v\}_{v \in V}$ is a partition of the set of targets J, *i.e.*, $\cup_{v \in V} J_v = J$ and $J_u \cap J_v = \emptyset$ for all $u \neq v$. A defender's strategy selects m precinct pairings and also the target where each resource team is deployed. The combined patrol from the pairing of precincts u and v can only be deployed to a target in $J_u \cup J_v$. For each edge $e = \{u, v\} \in E$ we define $J_e = J_u \cup J_v$. It follows that the set I of defender strategies can be expressed

as

$$I = \{(\mathbf{y}, \mathbf{w}) \in \{0,1\}^{|E|+|J|} : \mathbf{y} \in \mathcal{M}_m,$$
$$\sum_{j \in \cup_{v \in U} J_v} w_j \leq \sum_{e \in E(U) \cup \delta(U)} y_e \qquad U \subseteq V$$
$$\sum_{j \in J} w_j = m \qquad \}. \tag{2.14}$$

For $(\mathbf{y}, \mathbf{w}) \in I$, variable y_e indicates whether edge e is selected for a precinct pairing and w_j indicates whether target j is patrolled. The first condition enforces that $\mathbf{y}$ identifies a matching of size m in V. The second states that the number of possible targets to patrols inside each subset of nodes should not to be more than the possible pairs that can patrol them. The last condition enforces that m targets are protected. The two last conditions ensure that for each edge $\{u, v\}$ selected there is only one outpost selected inside J_u and J_v.

In this SSG, the defender selects $(\mathbf{y}, \mathbf{w}) \in I$ and an attacker of type $k \in K$, appearing with probability π^k, responds by selecting the target $j \in J$ to attack. This Stackelberg game can be represented by explicitly enumerating all strategies in I and using the MILP (D2) (or (MIP-G)) to obtain the optimal strategies. Unfortunately, this approach is computationally challenging as the resulting optimization problem considers one variable $x_{(\mathbf{y},\mathbf{w})}$ for each $(\mathbf{y}, \mathbf{w}) \in I$, which is an exponential number of variables in terms of the number of patrols to deploy. Below we introduce compact MILP formulations with a polynomial number of variables and exponentially many constraints separable in polynomial time.

The compact formulation is derived similarly to [19] and is based on the observation that if rewards are given by (2.12) and (2.13) then the utility of each player only depends on c_j, the coverage at a target $j \in J$. To introduce the compact formulation, given a probability distribution $x_{(\mathbf{y},\mathbf{w})}$ over the defender strategies $(\mathbf{y}, \mathbf{w}) \in I$, in addition to c_j we define z_e the coverage probability on edges $e \in E$ and $g_{e,j}$ the combined coverage over edge e and target $j \in J_e$ as follows:

$$c_j = \sum_{(\mathbf{y},\mathbf{w}) \in I \,:\, w_j = 1} x_{(\mathbf{y},\mathbf{w})} \qquad j \in J \tag{2.15}$$

$$z_e = \sum_{(\mathbf{y},\mathbf{w}) \in I \,:\, y_e = 1} x_{(\mathbf{y},\mathbf{w})} \qquad e \in E \tag{2.16}$$

$$g_{e,j} = \begin{cases} \displaystyle\sum_{(\mathbf{y},\mathbf{w}) \in I : y_e = 1, w_j = 1} x_{(\mathbf{y},\mathbf{w})} & \text{if } j \in J_e \\ 0 & \text{o/w} \end{cases} \qquad e \in E, j \in J\ . \tag{2.17}$$

Given a graph $G = (V, E)$ with V the set of precincts and E the feasible pairings between precincts, [4] introduce the following optimization problem as the equivalent compact version of (D2) when the set of defender strategies I is given by (2.14):

(BP)

$$\max_{c,z,q,s,f,g} \quad \sum_{k\in K} \pi^k f^k \tag{2.18}$$

$$\text{s.t.} \quad \mathbf{q}^{k\top}\mathbf{1} = 1,\ \mathbf{q}^k \in \{0,1\}^{|J|} \qquad \forall k \in K, \tag{2.19}$$

$$\mathbf{c}^\top \mathbf{1} = m,\ \mathbf{c} \in [0,1]^{|J|} \tag{2.20}$$

$$\mathbf{z}^\top \mathbf{1} = m,\ \mathbf{z} \in [0,1]^{|E|} \tag{2.21}$$

$$\sum_{e\in\delta(v)} z_e \le 1 \qquad \forall v \in V, \tag{2.22}$$

$$\sum_{e\in E(U)} z_e \le \frac{|U|-1}{2} \qquad \forall U \subseteq V,\ |U| \ge 3, |U| \text{ odd} \tag{2.23}$$

$$\sum_{e\in E: j\in J_e} g_{e,j} = c_j \qquad \forall j \in J \tag{2.24}$$

$$\sum_{j\in J_e} g_{e,j} = z_e \qquad \forall e \in E \tag{2.25}$$

$$f^k \le D^k(j|p)c_j + D^k(j|u)(1-c_j) + (1-q_j^k)\cdot M \qquad \forall j \in J, \forall k \in K, \tag{2.26}$$

$$0 \le s^k - A^k(j|p)c_j - A^k(j|u)(1-c_j) \le (1-q_j^k)\cdot M \qquad \forall j \in J, \forall k \in K \tag{2.27}$$

$$\mathbf{s}, \mathbf{f} \in \mathbb{R}^K,\ \mathbf{g} \in [0,1]^{|E||J|}\ . \tag{2.28}$$

Constraint (2.19) ensures that attacker $k \in K$ attacks a single target $j \in J$. Constraints (2.20) and (2.21) indicate that the defender uses all resources and precincts are paired without exceeding the number of resources deployed. Constraint (2.22) indicates that a precinct's contribution to a pairing cannot exceed 1. Constraints (2.23) correspond to the *blossom* or *odd set* inequalities, introduced in [14], and together with (2.21) and (2.22) enforce that the coverage probabilities on the edges belong to the convex hull of the polytope of matchings of size m. Constraints (2.24) and (2.25) enforce the conservation between marginal coverages in nodes and edges. Finally, Constraints (2.26) and (2.27)

ensure that $\mathbf{c}$ and $\mathbf{q}$ are mutual best responses. The objective function in our formulation, maximizes the defender's expected utility.

Here we present a new compact formulation (COMB) obtained from (MIP-G) when the set I is given by (2.14) by applying the coverage probabilities (2.15)-(2.17). In this formulation we use the transformation $t^k_{jl} = c_j q^k_l$ and that this implies $\sum_{l\in J} t^1_{jl} = c_j$ for any vector $\mathbf{q}$ feasible.

(COMB)

$$\max_{t,z,q,g} \quad \sum_{k\in K} \pi^k \sum_{j\in J} \left(D^k(j|p) t^k_{jj} + D^k(j|u)(q^k_j - t^k_{jj}) \right) \tag{2.29}$$

s.t. (2.19), (2.21), (2.22), (2.23), (2.25)

$$\sum_{l\in J} t^k_{lj} = m q^k_j \qquad \forall j \in J, \forall k \in K \tag{2.30}$$

$$0 \le t^k_{lj} \le q^k_j \qquad \forall l, j \in J, \forall k \in K \tag{2.31}$$

$$\sum_{j\in J} t^k_{lj} = \sum_{j\in J} t^1_{lj} \qquad \forall l \in J, \forall k \in K \tag{2.32}$$

$$\sum_{e\in E: j\in J_e} g_{e,j} = \sum_{l\in J} t^1_{jl} \qquad \forall j \in J \tag{2.33}$$

$$A^k(j|p) t^k_{jj} + A^k(j|u)(q^k_j - t^k_{jj}) - A(l|p) t^k_{lj} - A^k(l|u)(q^k_j - t^k_{lj}) \ge 0 \qquad \forall l, j \in J, \forall k \in K \tag{2.34}$$

$$\mathbf{t} \in [0,1]^{|J|^2 \times |K|},\ \mathbf{z} \in [0,1]^{|E|},\ \mathbf{g} \in [0,1]^{|E||J|}\ , \tag{2.35}$$

Constraints (2.30) states that the total coverage is equal to the amount of resources m. Constraints (2.31) establish the relationship between variables $\mathbf{t}$ and $\mathbf{q}$. Similarly to 2.8, expressions 2.32 come from the fact that $c_j = c_j \sum_{l\in J} q^k_{jl} = \sum_{j\in J} t^k_{jl}$ for each $k \in K$. Constraints (2.33) are equivalent to constraints (2.24) in (BP). Finally, the best response of each type of attacker is stated by Constraints (2.34). These last set of expressions, are equivalent to Constraints (2.27) in (BP) by multiplying by q^k_l and dropping out variables s^k through Fourier-Motzkin elimination. A proof of equivalence of (BP) and (COMB) is provided in Theorem 4 in Section 4.

In the following, we discuss how to translate solutions from (BP) and (COMB) to solutions of (D2) and the correctness of these formulations.

3. Algorithms

Now we provide algorithms to retrieve probability distributions over the original strategy space I from coverage distributions and a branch and cut implementation to separate constraints (2.23). As a consequence of this separation method we state complexity results for the problem of finding an SSE with paring resources.

3.1. Retrieving Pure Strategies from Coverage Probabilities

As we will see in Section 6, the compact formulations (BP) and (COMB) perform better than (D2) in terms of solution time. Anyway, the main issue with the compact formulations is that is not trivial how to implement a coverage vector in terms of feasible strategies of the set I. In other words, we need a way to retrieve a vector $\mathbf{x}$ feasible for (D2) given $(\mathbf{z}, \mathbf{c})$ feasible for (BP) or (COMB). In the real application, it means that the translation from frequencies of coverage to a real plan is needed.

We show a procedure to face this issue. To build $\mathbf{x}$ we first recognize that by constraints (2.22) and (2.23), the vector $\mathbf{z} \in$ convex hull$(\mathcal{M}_m)$. Therefore, there is a finite set of integer m-matchings $M_z \subseteq \mathcal{M}_m$ such that we can express $\mathbf{z} = \sum_{\mathbf{y} \in M_z} \lambda_{\mathbf{y}} \mathbf{y}$, where $\lambda_y \geq 0$ for all $y \in M_z$ and $\sum_{\mathbf{y} \in M_z} \lambda_{\mathbf{y}} = 1$. Each vector $\lambda_{\mathbf{y}}$ is the probability of playing a pure action involving pairings given by matching $\mathbf{y}$. Algorithm 1 is used to decompose these frequencies into a probability distribution over set I.

Algorithm 1 Procedure to identify mixed strategy $\mathbf{x}$ given feasible vectors $\mathbf{z}$, $\mathbf{c}$, $\mathbf{g}$

Step 1. For every edge $e \in E$ consider a column of height 1. Divide this $1 \times |E|$ rectangle in horizontal segments, one for each integer matching $\mathbf{y} \in M_z$, with each segment of width $\lambda_{\mathbf{y}}$. For each segment, corresponding to matching $\mathbf{y}$, block out the areas corresponding to the edges that are not used in $\mathbf{y}$ (i.e.'NO' in Figure 2).

Step 2. For every edge $e \in E$, further subdivide the total area in its column that is not blocked out into smaller rectangles using the values of g_{ej} for all $j \in J_e$. Since $\sum_{j \in J_e} g_{ej} = z_e = \sum_{\mathbf{y} \in M_z} \lambda_{\mathbf{y}} y_e$, from (2.25) and the decomposition of $\mathbf{z}$, this partition uses all the area in the column of e that was not blocked out.

Step 3. Define $\mathbf{x}$ by extending each rectangle line into a horizontal line crossing all columns. The area between two horizontal lines represents a defender strategy. This area is contained in a matching $\mathbf{y}$, that identifies the paired precincts, and for each edge e in the matching ($y_e = 1$), the area includes part of some g_{ej_e}, identifying the protected target j_e. Let the width of this horizontal line be $x_{(\mathbf{y},\mathbf{w})}$, where $\mathbf{w} \in \{0,1\}^{|J|}$ is the indicator vector of the protected targets $\{j_e\}_{\{e:y_e=1\}}$. For the rest of $(\mathbf{y}, \mathbf{w}) \in I$ not included in this procedure, set $x_{(\mathbf{y},\mathbf{w})} = 0$

Figure 2 describe how this algorithm works. For the sake of clarity, we provide a numerical example of Algorithm 1 in Appendix A. The correctness of Algorithm 1 is stated in Lemma 1 and its proof is

provided in Appendix B.

Lemma 1. *Given a set of matchings of size M_z and vectors $\mathbf{z}, \mathbf{c}, \mathbf{g}$, Algorithm 1 returns a vector x satisfying (2.2), (2.15), (2.16) and (2.17).*

This algorithm is an important step to prove the equivalence of the models stated in Section 4.

3.2. Branch and Cut Implementation

Finding a solution for the compact formulations (BP) and (COMB) remains difficult since the blossom inequalities, present in both formulations in (2.23) are exponentially many. To face this large number of constraints we implement a branch and cut scheme where blossom inequalities are generated on the fly in a cut generation algorithm. The branch and cut scheme follows the approach introduced in [23] and was implemented using the Lazy Constraint Callback in CPLEX 12.7.

This procedure either detects whether a vector $\mathbf{z}$ satisfies the blossom inequality constraints or identifies the most violated one. In this procedure, the problem of detecting if a vector $\mathbf{z}$ satisfies all the set of blossom inequalities is reduced to a global min-cut set problem with odd cardinality in a related network. This problem can be solved in polynomial time with the Gomory-Hu algorithm. A simple implementation of the Gomory Hu algorithm, given by [15], is used in our implementation. Details in the implementation are provided in Appendix D.

3.3. Complexity Results

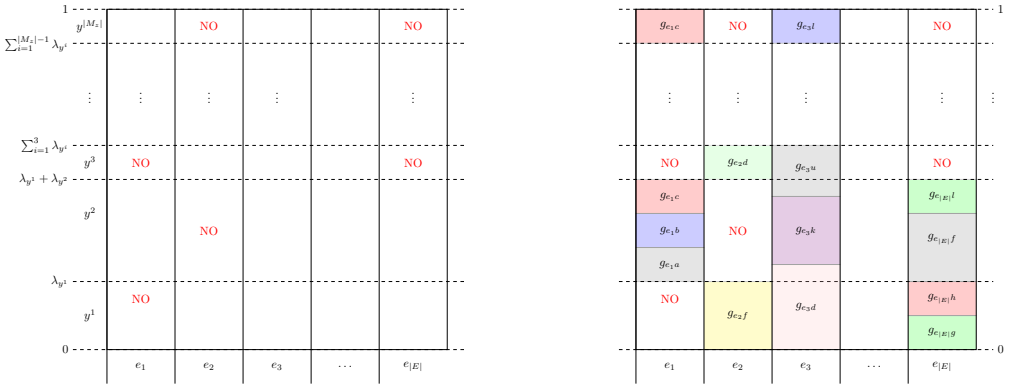
The methodology in [23] gives a polynomial time algorithm to separate (2.23). We use this algorithm to prove that this problem can be solved in polynomial time when $|K| = 1$. Then, we prove that this problem is NP-hard with an arbitrary number of followers.

Theorem 1. *Let J be he set of possible targets to protect and I the set of pure strategies for the Defender as in (2.14) respectively. Consider payoffs as in (2.12) and (2.13). The problem of finding a SSE in this setting with one single follower type can be solved in polynomial time.*

Proof. For a fixed target j, the problem of solving the best strategy for the follower to incentive the attacker to attack j is the following linear program:

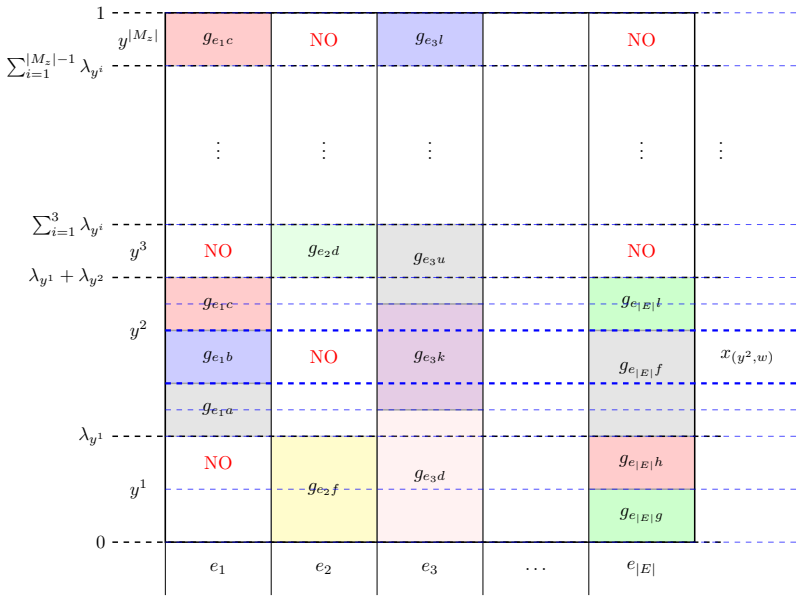
$$
\begin{aligned}
(\mathrm{LP}_j) \quad \max_{c} \quad & D(j|p)c_j + D(j|u)(1-c_j) \\
\text{s.t.} \quad & (2.19),\ (2.21),\ (2.22),\ (2.23),\ (2.24),(2.25), \\
& A(j|p)c_j + A(j|u)(1-c_j) \geq A(l|p)c_l + A(l|u)(1-c_l) \qquad \forall l \in J, \neq j
\end{aligned}
$$

where we omit the dependence of k of the payoffs. This is a bounded LP, and for some targets infeasible. Anyway, the set of targets where the problem is feasible is non-empty.



a) Step 1

b) Step 2



c) Step 3

Figure 2: Construction to identify mixed strategy $\mathbf{x}$ given feasible vectors $\mathbf{z}, \mathbf{c}, \mathbf{g}$. Here matching $\mathbf{y}^2$ includes edges $e_1, e_3, \ldots, e_{|E|}$. A constant width horizontal line (thicker) can be identified using portions of $g_{e_1b}, g_{e_3k}, \ldots, g_{|E|f}$. This shaded horizontal line corresponds to using precinct parings according to matching $\mathbf{y}^2$ and protecting targets $\mathbf{w} = \{b, k \ldots, f\}$, whose width we label as $x_{(\mathbf{y}^2,\mathbf{w})}$. For the sake of clarity, in step 3 we represent the rest of the strategies in shaded dashed lines.

Each of this linear programs has a polynomial number of variables and constraints that are separable in polynomial time due to [23]. Then each (LP_j) can be solved in polynomial time.

Then, by picking j which (LP_j) has the maximum objective value, the corresponding optimal solution is a SSE. The complexity of this algorithm is given by the complexity of solving the $|J|$ LPs. Then, the result follows. □

Theorem 2. *Let J the set of possible targets to protect and I the set of pure strategies for the Defender as in (2.14) respectively. Consider payoffs as in (2.12) and (2.13). The problem of finding a SSE in this setting with an arbitrary set of followers is NP-Hard.*

Proof. Consider an instance of a SSG which it is known to be NP-hard [20]. Now consider the following instance of the problem of pairing resources. For each target j, create a vertex v_j and a vertex copy u_j. Also, define the set of edges as $E = \{\{v_j, u_j\} : j \in J\}$, one target for each node, and the rest of the data remains the same. By solving the problem with pairing resources a solution of the SSG is obtained. Then, the problem of Pairing resources is also NP-hard. □

4. Problem Equivalence

In order to prove the correctness of the formulations we first state a useful property that gives us the relationship between $\mathbf{y}$ and $\mathbf{w}$ for each $(\mathbf{y}, \mathbf{w}) \in I$. We provide the proof of this property on Appendix C.

Lemma 2. *Given $(\mathbf{y}, \mathbf{w}) \in I$ then the following holds*

$$w_j = 1 \Rightarrow \sum_{e \in E : j \in J_e} y_e = 1 \tag{4.1}$$

$$y_e = 1 \Rightarrow \sum_{j \in J_e} w_j = 1. \tag{4.2}$$

We now present the equivalence proofs to show that (BP) and (COMB) are correct formulations for computing the SSE of the problem. First, we show that (BP) is a formulation of the SSE for this problem by showing the equivalence with (D2) in a game with payoff structure given by (2.12) and (2.13). Then, we provide the equivalence result of (COMB) and (BP).

Theorem 3. *Let the set I be given by (2.14) and the reward matrices by (2.12) and (2.13). Then Problem (BP) given by (2.18)-(2.28) is equivalent to (D2) given by (2.1)-(2.6).*

Proof. We begin with a solution $(\mathbf{x}, \mathbf{q}, \mathbf{s}, \mathbf{f})$ feasible for (D2). Given this $\mathbf{x}$ we define $\mathbf{c} \in [0,1]^{|J|}$, $\mathbf{z} \in [0,1]^{|E|}$ and $\mathbf{g} \in [0,1]^{|E||J|}$ as in (2.15), (2.16), and (2.17), respectively. We now show that this $(\mathbf{c}, \mathbf{z}, \mathbf{q}, \mathbf{s}, \mathbf{f}, \mathbf{g})$ is feasible for (BP) and has the same objective value. Indeed, the objective function

(2.18) and constraints (2.19) in (BP) are the same as (2.1) and (2.3) in (D2). In addition constraints (2.28) are satisfied from the definition of $\mathbf{c}$, $\mathbf{z}$ and $\mathbf{g}$ and the fact that $\mathbf{x}$ is a probability distribution.

Since the utilities are given by (2.12) and (2.13), we have that

$$\begin{aligned}\sum_{(\mathbf{y},\mathbf{w})\in I} R^k_{(\mathbf{y},\mathbf{w})j} x_{(\mathbf{y},\mathbf{w})} &= \sum_{(\mathbf{y},\mathbf{w})\in I:w_j=1} R^k_{(\mathbf{y},\mathbf{w})j} x_{(\mathbf{y},\mathbf{w})} + \sum_{(\mathbf{y},\mathbf{w})\in I:w_j=0} R^k_{(\mathbf{y},\mathbf{w})j} x_{(\mathbf{y},\mathbf{w})} \\ &= D^k(j|p) \sum_{(\mathbf{y},\mathbf{w})\in I:w_j=1} x_{(\mathbf{y},\mathbf{w})} + D^k(j|u) \left(1 - \sum_{(\mathbf{y},\mathbf{w})\in I:w_j=1} x_{(\mathbf{y},\mathbf{w})}\right) \\ &= D^k(j|p)c_j + D^k(j|u)\,(1-c_j)\ .\end{aligned}$$

This equality is used to conclude that (2.4) implies (2.26). Analogously for the attackers utility we have that (2.5) implies (2.27).

From the definition of $g_{e,j}$ we can write

$$\begin{aligned}\sum_{j\in J_e} g_{e,j} = \sum_{j\in J_e} \sum_{\substack{(\mathbf{y},\mathbf{w})\in I \\ y_e=1,\, w_j=1}} x_{(\mathbf{y},\mathbf{w})} &= \sum_{j\in J_e} \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} y_e w_j \\ = \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} y_e \sum_{j\in J_e} w_j &= \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} y_e = z_e\ .\end{aligned}$$

Where the next to last equality follows from (4.2). This shows that (2.25) is satisfied. Similarly expanding $\sum_{e\in E:j\in J_e} g_{e,j}$ and now using (4.1) gives that (2.24) is satisfied. In the same logic it follows that

$$\sum_{e\in E} z_e = \sum_{j\in J} c_j = \sum_{e\in E}\sum_{j\in J_e} g_{e,j} = \sum_{e\in E}\sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} y_e = \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} \sum_{e\in E} y_e = m\ .$$

This proves that (2.21) is satisfied. Finally we observe above that the vector $z \in [0,1]^{|E|}$ satisfies $\mathbf{z} = \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})}\mathbf{y}$ where $\mathbf{x}$ is a probability distribution over I and the vectors $\mathbf{y}$ correspond to m-matchings in the set V. Therefore $\mathbf{z} \in$ convex hull$(\mathcal{M}_m)$, which implies that $\mathbf{z}$ satisfies constraints (2.22) and (2.23) that characterize the m-matching polytope, completing the first part of the proof.

We now consider a solution $(\mathbf{c},\mathbf{z},\mathbf{q},\mathbf{s},\mathbf{f},\mathbf{g})$ feasible for (BP) and construct a solution $(\mathbf{x},\mathbf{q},\mathbf{s},\mathbf{f})$ that is feasible for (D2) with the same objective function. Variables $\mathbf{q}$, $\mathbf{s}$ and $\mathbf{f}$ are the same, this shows that the objective (2.1) and constraint (2.3) in (D2) are satisfied as they correspond to (2.18) and (2.19) respectively. To build $\mathbf{x}$ we first recognize that by constraints (2.22) and (2.23), the vector $\mathbf{z} \in$ convex hull$(\mathcal{M}_m)$. Therefore, there is a finite set of integer m-matchings $M_z \subseteq \mathcal{M}_m$ such that we can express $\mathbf{z} = \sum_{\mathbf{y}\in M_z} \lambda_{\mathbf{y}}\mathbf{y}$, where $\lambda_{\mathbf{y}} \geq 0$ for all $\mathbf{y} \in M_z$ and $\sum_{\mathbf{y}\in M_z} \lambda_{\mathbf{y}} = 1$. Given this decomposition of $\mathbf{z}$, by applying Algorithm 1, we obtain a feasible $\mathbf{x}$ for (D2). By Lemma 1, $\mathbf{x}$ satisfies (2.15), then the objective value in (D2) is equal to the solution for (BP). The result follows.

□

Theorem 4. *The Formulation (BP) given by (2.18)-(2.28) is equivalent to (COMB) given by (2.29)-(2.35).*

Proof. Consider first $(\mathbf{c}, \mathbf{z}, \mathbf{q}, \mathbf{s}, \mathbf{f}, \mathbf{g})$ the optimal solution of (BP). Defining variables $t^k_{lj} = c_l q^k_j$, we show that $(\mathbf{t}, \mathbf{z}, \mathbf{q}, \mathbf{g})$ is feasible for (COMB) with the same expected reward.

Note that, the expected utility of (BP) can be written as

$$\sum_{k\in K} \pi^k \sum_{j\in J} [D^k(j|p)c_j + D^k(j|u)(1-c_j)]q^k_j = \sum_{k\in K} \pi^k \sum_{j\in J} [D^k(j|p)t^k_{jj} + D^k(j|u)(q^k_j - t^k_{jj})],$$

so the solution $(\mathbf{t}, \mathbf{z}, \mathbf{q}, \mathbf{g})$ has the same objective value. Constraints (2.19), (2.21), (2.22), (2.23), (2.25) are the same in both problems, so they are satisfied by this solution. Constraints (2.20) in (BP) imply constraints (2.30) in (COMB). Given that $c_l \in [0,1]$ and q^k_j is binary, implies that $\mathbf{t}$ satisfies $0 \le t^k_{lj} \le q^k_j$ satisfying constraints (2.31). Note that $\sum_{j\in J} t^k_{lj} = \sum_{j\in J} c_l q^k_j = c_l$, is independent of k, therefore $\mathbf{t}$ satisfies (2.32). By the same argument, constraints (2.33) are also satisfied.

Finally, if the follower type k best response is to attack j, we have that $q^k_j = 1$. Then

$$\begin{aligned} s^k &= A^k(j|p)c_j + A^k(j|u)(1-c_j)]q^k_j \\ &= A^k(j|p)t^k_{jj} + A^k(j|u)(q^k_j - t^k_{jj}) \\ &\ge A(l|p)t^k_{lj} + A^k(l|u)(q^k_j - t^k_{lj}) \quad l \in J,\, l \ne j. \end{aligned}$$

When $q^k_j = 0$ constraints (2.34) are trivially satisfied. At this point we show that the optimal objective value of (BP) is upper bounded by the optimal objective value of (COMB).

Now we pick $(\mathbf{t}, \mathbf{z}, \mathbf{q}, \mathbf{g})$, the optimal solution of (COMB). By taking $c_j = t^1_{jj}$ for each $j \in J$, $s_k = \sum_{j\in J}[A^k(j|p)t^k_{jj} + A^k(j|u)(q^k_j - t^k_{jj})]$ and $f^k = \sum_{j\in J} D^k(j|p)t^k_{jj} + D^k(j|u)(q^k_j - t^k_{jj})$ for each $k \in K$, we state $(\mathbf{c}, \mathbf{z}, \mathbf{q}, \mathbf{s}, \mathbf{f}, \mathbf{g})$ is feasible to (BP) and it has the same expected reward. By definition of f^k, the objective value of (COMB) has the same value in (BP). It is easy to see that constraints (2.20) and (2.24) are satisfied by this new solution.

Given the definition of f^k, (2.26) is tight when $q^k_j = 1$ and given that $M \gg 1$ is also satisfied when $q^k_j = 0$. Finally, the definition of s^k implies that

$$\begin{aligned} s_k &= \sum_{j\in J} [A^k(j|p)t^k_{jj} + A^k(j|u)(q^k_j - t^k_{jj})] \\ &= \sum_{j\in J} [A^k(j|p)c_j + A^k(j|u)(1-c_j)]q^k_j. \end{aligned}$$

Then, using that $\mathbf{q}^k\mathbf{1} = 1$, we note that (2.27) is the linearization of the last equation. Then $(\mathbf{c}, \mathbf{z}, \mathbf{q}, \mathbf{s}, \mathbf{f}, \mathbf{g})$ is feasible for (BP). Then, the result follows. □

5. Sampling a strategy

A strategy that can be implemented by security forces determines which pure strategy is done in each shift. In this section we describe how we use the optimal solutions to (BP) or (COMB), i.e. vectors $\mathbf{z}^*$, $\mathbf{c}^*$, to sample strategies $(\mathbf{y}, \mathbf{w})$ from I.

A first approach is to sample from I according to the optimal strategy $\mathbf{x}^*$. This can be achieved by constructing $\mathbf{x}^*$ from $\mathbf{z}^*$, $\mathbf{c}^*$ by Algorithm 1. This procedure requires that a fractional m-matching $\mathbf{z}^*$ is decomposed as a convex combination of pure matchings of size m. An algorithm to construct this decomposition is presented next. We finish this section with an alternative sampling method that uses solutions $(\mathbf{z}^*, \mathbf{c}^*)$ directly.

5.1. Decomposing $\mathbf{z}$ *in pure matchings of size* m

Given a vector $\mathbf{z}$ satisfying (2.21)-(2.23) we want to build a set $M_{\mathbf{z}} \subseteq \mathcal{M}_m$ of matchings of size m and a set of weights $\{\lambda_{\mathbf{y}}\}_{\mathbf{y}\in\mathcal{M}_m}$ such that the vector $\mathbf{z}$ can be written as a convex combination of elements in $M_{\mathbf{z}}$. This can be achieved by using a Dantzig-Wolfe approach, [10]. The master problem can be stated as:

$$
\begin{aligned}
\text{(MP)} \quad \min \quad & \sum_{e\in E} Y_e + \Lambda && && (5.1)\\
& Y_e + \sum_{\mathbf{y}^i\in\mathcal{M}_m} \lambda_{\mathbf{y}^i} y^i_e = z_e && \forall e \in E && (5.2)\\
& \sum_{\mathbf{y}\in\mathcal{M}_m} \lambda_{\mathbf{y}^i} + \Lambda = 1 && && (5.3)\\
& \lambda_{\mathbf{y}} \geq 0 && \forall \mathbf{y} \in \mathcal{M}_m && (5.4)\\
& Y_e, \Lambda \geq 0 && \forall e \in E, && (5.5)
\end{aligned}
$$

where $\lambda_{\mathbf{y}^i}$ is the weight of matching $\mathbf{y}^i$. Variables Y_e and Λ represent auxiliary variables to be minimized. Here, $\mathbf{y}^i_e$ takes value 1 when the edge e is included in the matching $\mathbf{y}^i$. Let γ_e and σ be the optimal dual variable associated to constraint (5.2) and (5.3) respectively. Then, the reduced costs of any column/matching is given by:

$$r_{\mathbf{y}} = 0 - \sum_{e\in E} \gamma_e y_e - \sigma.$$

The problem of adding a new column to a restricted master problem (i.e. only consider a subset of columns $M_{\mathbf{z}} \subset \mathcal{M}_m$) can be stated as a maximum weight matching of size m with weights $\{\gamma_e\}_{e\in E}$. If a matching y has a weight greater than $-\sigma$ it is added to $M_{\mathbf{z}}$ and the restricted master is re-optimized. The algorithm stops when no new column is added (and the objective (5.1) is equal to zero), in which case $\mathbf{z}$ can be written as a convex combination of matchings in $M_{\mathbf{z}}$.

Given that the column generator problem is a maximum weighted matching of size m, it can be solved in polynomial time. Then, the problem of decomposing $\mathbf{z}$ in a convex combination of pure matching can be solved in polynomial time. Furthermore, Algorithm 1 can be performed in $\mathcal{O}(|E|^2|J|)$. Thus, the next result follows:

Theorem 5. *Given the vectors* $(\mathbf{z}, \mathbf{c}, \mathbf{g})$ *satisfying (2.20) - (2.25), a vector of probabilities* $\mathbf{x}$ *over* I *satisfying (2.15) - (2.17) can be retrieved in polynomial time.*

5.2. An approximate sampling method

Finding the convex combination decomposition of $\mathbf{z}$ can be computationally expensive, even with the polynomial solvability result above. We now propose a computationally efficient, but approximate, two stage sampling method that generates points $(\mathbf{y}, \mathbf{w}) \in I$ directly from $\mathbf{z}$ and $\mathbf{c}$. In Section 6, we investigate how fast and accurate this approximate sampling method is.

Given $\mathbf{z}$, we seek to select m edges that form a matching. We do this by iteratively sampling edges without replacement to build the set $\mathcal{E}$ and solving the maximum weight m-matching problem

$$\begin{aligned} \text{Max} \quad & \sum_{e \in \mathcal{E}} z_e x_e \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{M}_{\mathbf{m}}(\mathcal{E}) \ , \end{aligned}$$

where $\mathcal{M}_m(\mathcal{E})$ is the set of m-matchings using only edges in $\mathcal{E}$. To sample edges we consider a fixed index order for the set E. At a given iteration, let $\hat{m} = \sum_{e \in E \setminus \mathcal{E}} z_e$ be the sum of $\mathbf{z}$ over edges not yet sampled and generate a number between 0 and $\hat{m}$ uniformly. We select the edge in $E \setminus \mathcal{E}$ that causes the partial sum of $\mathbf{z}$, according to the fixed order, meet or exceed the generated number for the first time. This edge is then added to $\mathcal{E}$.

Solving the maximum weight m-matching problem can have two possible outcomes: the problem is either infeasible and there are no matchings of size m with edges in $\mathcal{E}$ or the problem returns an optimal solution. If the problem is infeasible we continue iterating and sampling more edges for the set $\mathcal{E}$. If the problem returns an optimal matching we keep this as the sampled matching $\mathbf{y}$. Note that this algorithm will produce a matching in at most $|E|$ iterations.

We generate the targets protected by sampling according to the normalized target coverage probability, defined below, for the precincts covered by the matching $\mathbf{y}$ found in the previous step. That is, we use the discrete probability

$$\bar{c}_j = \frac{c_j}{\sum_{k \in J_e} c_k} \qquad \forall j \in J_e \text{ s.t. } y_e = 1$$

to sample a single target for every $e \in E$ such that $y_e = 1$. This generates the protected targets that agree with the selected matching, giving the point $(\mathbf{y}, \mathbf{w}) \in I$. We refer to this method of generating solutions in I as the Approximate Sampling (AS) method.

6. Computational Results

In order to obtain efficient patrolling strategies for real world security problems it is necessary to develop algorithms that are able to handle large instances. Therefore, in this section we evaluate the computational efficiency of the compact formulations (BP) and (COMB). Given an optimal patrolling strategy, it is important in a real security situation to quickly analyze different deployment strategies. We, therefore, study the effectiveness of the alternative sampling procedure presented in Section 5.2.

All the experiments in this section were programmed in Python 3.5 using CPLEX 12.7.

6.1. Computational Performance

We compare the performance of the general Stackelberg formulations (D2) and (MIP-G) to the Stackelberg security formulations developed specifically for the combined defender patrol problem (COMB) and (BP), and their branch and cut schemes described in Section named in 3.2. We refer to the branch and cut versions as (COMB_BC) and (BP_BC).

We consider randomly generated connected graphs with n nodes, a set of edges with an average node degree of 3 and m security resources. Further, we consider four targets within each node and three types of attackers. The payoff values are randomly generated such that penalties for the defender and each attacker are between - 5 and 0 and rewards for the defender and each attacker are between 5 and 10. We scale instances by increasing n and m.

We first focus on instances with two pairings (involving four nodes, $m = 2$) and $n \in \{5, 6, 7, \ldots, 22\}$. We solve five random instances for each problem size considered and report expected results. In Figure 3, we show, on the left, the time (in seconds) it takes to solve the instances as a function of the number of nodes in the graph, and on the right a performance profile, indicating the number of blossom inequalities added by the branch and cut procedure against the total percentage of instances solved.

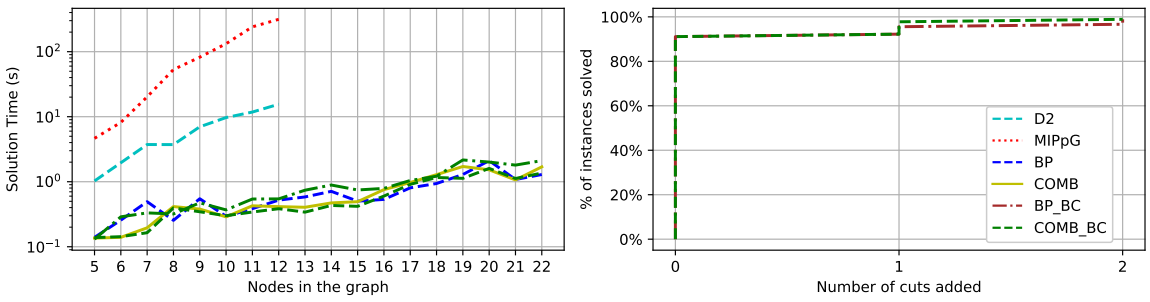


Figure 3: Instances with 2 security resources in graphs of up to 22 nodes

We note that the general Stackelberg formulations (MIP-G) and (D2) can only solve small instances, at most with 8 and 11 nodes, respectively. We therefore do not consider formulations (MIP-G) and (D2) in the rest of our analysis. The compact formulations take less than three seconds to solve on average and the number of cuts generated by the branch and cut versions is small (at most two cuts are generated and 90% of instances require no cuts to find the optimal solution).

Our next set of experiments consider larger instances, with $n \in \{25, 30, 35, 40\}$ and $m = 4$ pairings. We present performance profile graphs for these instances in Figure 4. We observe that the branch and cut variants take less overall solution time and time to solve the LP relaxation, explore fewer nodes in the branch and bound tree and have a smaller MIP gap than (BP) and (COMB). This can be explained by the observation from Figure 3 that few blossom inequalities are required to solve these problems.

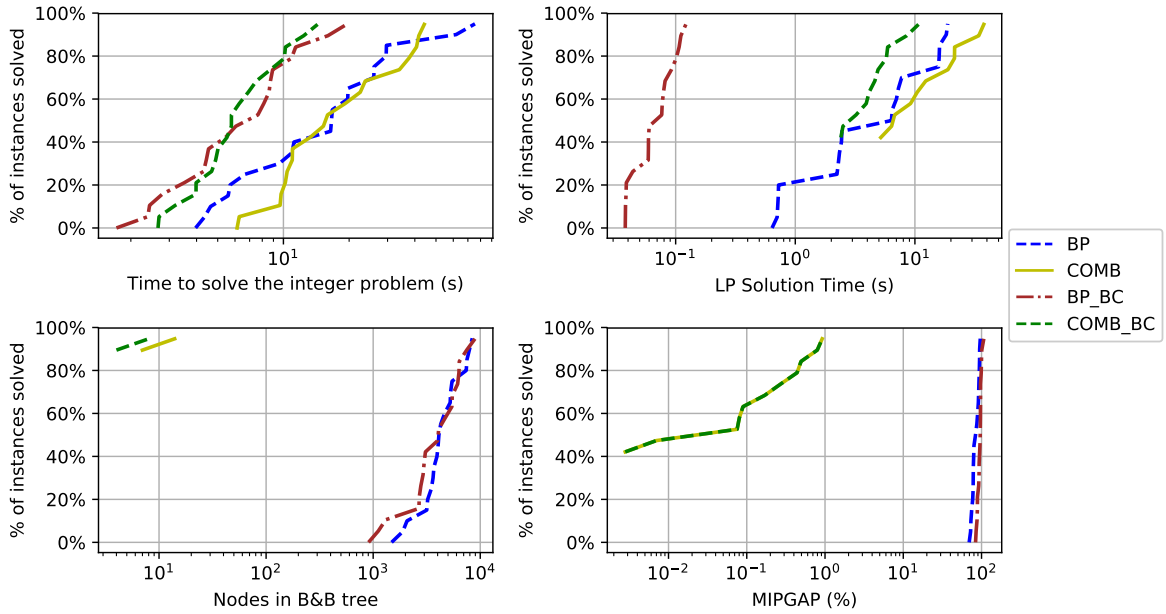


Figure 4: Performance profile graphs for instances with 4 security resources in graphs of up to 40 nodes

We further scale the instances to compare the efficiency of (BP_BC) and (COMB_BC). We consider graphs with $n \in \{30, 40, \ldots, 80\}$ and $m = 10$ security resources. number of cuts added by the branch and cut procedure against the percentage of problems solved. In Figure 5 we present the performance profile graphs for these instances.

These results show that solution times are comparable for both formulations, with a slight preference for (BP_BC) for larger instances. Here we observe the tradeoff between number of iterations and work per iteration. Formulation (COMB_BC) has a linear relaxation that takes longer to solve

but that is compensated with achieving a smaller MIP gap, thus requiring less nodes in the branch and bound tree.

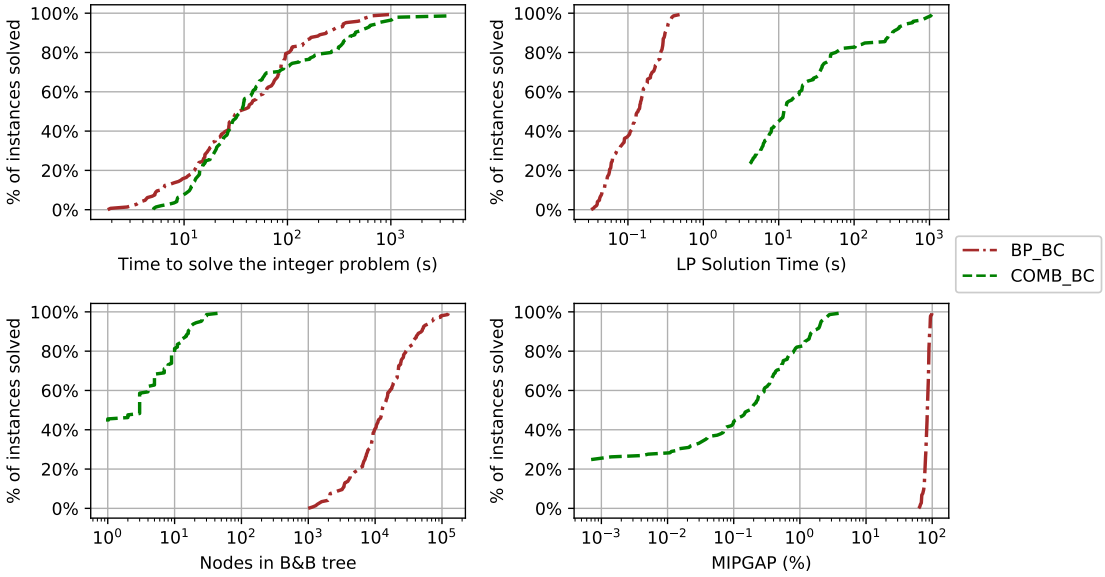


Figure 5: Performance profile graphs for instances with 10 security resources in graphs of up to 80 nodes

Further increasing problem size we observe in Figure 6 that the (BP_BC) formulation is able to solve instances on graphs with $n = 200$ and $m = 25$ security resources within a 1 hour time limit. We note that there is a trade-off between number of nodes and resources for the largest problems that can be solved, with an instance on $n = 150$ and $m = 15$ also taking about an hour to solve.

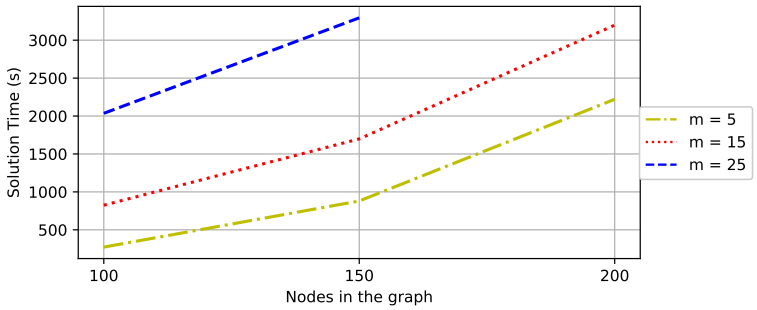


Figure 6: Solution times for (BP_BC) in instances with $n = 100, 150, 200$ and $m = 5, 15, 25$

6.2. Performance of sampling methods

Here we evaluate the computational time and accuracy of the Approximate Sampling (AS) method presented in Section 5.2. The sampling method obtains implementable patrolling strategies (solutions

$(\mathbf{y}, \mathbf{w}) \in I$) from the solution $(\mathbf{z}, \mathbf{c})$ provided by the compact formulation of the combined defender security problem.

In Table 1 we show the computational times required by this heuristic method and the exact sampling method based on column generation (Section 5.1). The approximate sampling method can determine an implementable strategy in less than a second, independent of problem size. On the other hand, the computational time for the exact method increases dramatically with the size of the problem.

n/m	25/3	50/3	100/3	25/10	50/10	100/10
AS [s]	0.028	0.030	0.026	0.184	0.064	0.045
EM [s]	2.222	8.345	177.133	6.426	55.008	1176.458

Table 1: Computational time (seconds) of Approximate Sampling Method (AS) and Exact Method (EM).

To evaluate the accuracy of the approximate sampling method we compare the resulting empirical coverage $(\hat{\mathbf{z}}, \hat{\mathbf{c}})$ obtained by repeatedly applying AS to the coverage solution $(\mathbf{z}, \mathbf{c})$ used to execute AS. This means that after conducting N approximate samples we let $\hat{c}_j = \frac{\text{Times target } j \text{ is selected}}{N}$ for each target $j \in J$. The value $\hat{z}_e$ is defined analogously. It is not possible to do this comparison with the corresponding strategies $\mathbf{x}$ over the set of combined actions I because there are multiple such strategies for every given coverage solution $(\mathbf{z}, \mathbf{c})$. To measure the error between $\mathbf{c}$ and our approximate $\hat{\mathbf{c}}$ we use the infinity norm $\|\mathbf{c} - \hat{\mathbf{c}}\|_\infty = \max_{\mathbf{j} \in \mathbf{J}} |\mathbf{c_j} - \hat{\mathbf{c}}_\mathbf{j}|$.

We consider a sampling size of $N = 1000$ over instances with $n \in \{25, 50, 100, 250\}$ nodes and $m \in \{3, 10\}$ security resources. For each instance size we generate 30 estimated $\hat{\mathbf{c}}$ and plot the results as box diagrams as shown in Figure 7.

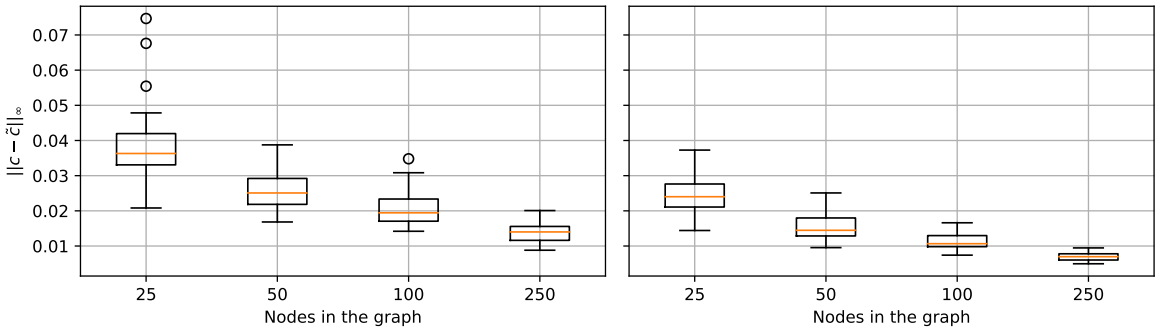


Figure 7: Total variation distance between $\mathbf{c}$ and $\hat{\mathbf{c}}$ over instances of different size for $m = 3$ and $m = 10$ respectively.

First, we notice that for larger instances the distance between $\mathbf{c}$ and $\hat{\mathbf{c}}$ decreases and so does the

variance of this distance. Second, even for the smallest instance considered the maximum error is not bigger than 0.06. The approximate sampling method proposed is a fast and accurate way to obtain sampled strategies close to optimal ones, in particular for large instances. Analogous results are obtained for the Total Variation distance between $\mathbf{z}$ and the approximate $\hat{\mathbf{z}}$.

7. Conclusions

In this article, we studied a special type of SSG played on a network. In this game, a defender combines resources to patrol a set of targets. The resulting combinatorial structure of the leader's set of actions makes solving real sized instances of this Stackelberg Game computationally challenging. The objective of this work is to develop efficient solution methods to solve this problem for real security instances. Furthermore, to patrol the set of targets according to the optimal strategy found requires, in addition, to being able to efficiently sample this optimal mixed strategy. We propose a formulation, (COMB), that represents the set of mixed strategies in a compact form via marginal probabilities in the pairs of resources that team-up and the targets covered. We propose a method to retrieve an implementable strategy given the optimal marginals and we prove the validity of our formulation. To scale-up the instances that could be optimized by solvers, we use a cut generation algorithm. We also provide complexity results by showing a polynomial time algorithm to solve the case when $|K| = 1$ and the NP-hardness for an arbitrary set of followers. We further provide an alternative sampling method to recover an implementable defender strategy given the optimal coverage distributions.

Our computational results show that the the branch and cut variants are computationally efficient, with the (BP-BC) capable of solving instances with 600 targets (or 150 nodes) protected by 25 combined resources in about one hour. The approximate sampling method proposed is able to reduce the sampling computational times several orders of magnitude incurring in a small error that is reduced as the problem size increases.

In future work we aim to extend our model and methodology to incorporate more realistic security situations, such as the case where more than two resources are combined to form a patrol there are different resource capabilities and time schedules.

Acknowledgments

Víctor Bucarey and Martine Labbé have been partially supported by the Fonds de la Recherche Scientifique - FNRS under Grant(s) no PDR T0098.18. Víctor Bucarey was also partially funded by the CONICYT PFCHA/DOCTORADO NACIONAL/2013 - 21130556. Fernando Ordóñez acknowledges the support from Instituto Sistemas Complejos de Ingeniería (ISCI) through CONICYT - PIA/BASAL AFB180003 and from FONDECYT-Chile, grant No. 1171419.

Bibliography

[1] Aduanas de Chile (2016). ¿Qué tributos deben pagar las importaciones? [Which taxes must the imports pay?]. https://www.aduana.cl/importaciones-de-productos/aduana/2007-02-28/161116.html.

[2] Alpern, S., Morton, A., & Papadaki, K. (2011). Patrolling games. *Operations research*, *59*, 1246–1257.

[3] Bracken, J., & McGill, J. T. (1973). Mathematical programs with optimization problems in the constraints. *Operations Research*, *21*, 37–44.

[4] Bucarey, V., Casorrán, C., Figueroa, O., Rosas, K., Navarrete, H., & Ordóñez, F. (2017). Building real stackelberg security games for border patrols. In *Decision and Game Theory for Security: 8th International Conference, GameSec 2017*.

[5] Canty, M., Rothenstein, D., & Avenhaus, R. (2001). A sequential attribute sampling inspection game for item facilities. *Naval Research Logistics (NRL)*, *48*, 496–505.

[6] Casorrán, C., Fortz, B., Labbé, M., & Ordóñez, F. (2019). A study of general and security stackelberg game formulations. *European Journal of Operational Research*, *278*, 855 – 868. doi:`https://doi.org/10.1016/j.ejor.2019.05.012`.

[7] CEPAL (2000). Costo económico de los delitos, niveles de vigilancia y políticas de seguridad ciudadana en las comunas del Gran Santiago [Economic cost of crime, surveillance levels and citizen security policies in the communes of Santiago City]. http://www.cepal.org/es/publicaciones/7258-costo-economico-de-los-delitos-niveles-de-vigilancia-y-politicas-de-seguridad.

[8] Conitzer, V., & Sandholm, T. (2006). Computing the optimal strategy to commit to. In *Proceedings of the 7th ACM conference on Electronic commerce* (pp. 82–90). ACM.

[9] Council of the EU (2016). European border and coast guard: final approval. http://www.consilium.europa.eu/en/press/press-releases/2016/09/14-european-border-coast-guard/. Retrieved on 02/2017.

[10] Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations research*, *8*, 101–111.

[11] Department of Homeland Security (2017). Border security. https://www.dhs.gov/border-security. Retreived on 02/2017.

[12] Dimitrov, N. B., & Morton, D. P. (2013). Interdiction models and applications. In *Handbook of Operations Research for Homeland Security* (pp. 73–103). Springer.

[13] Dreiding, R., & McLay, L. A. (2013). An integrated screening model for screening cargo containers for nuclear weapons. *European Journal of Operational Research*, *230*, 181–189.

[14] Edmonds, J. (1965). Maximum matching and a polyhedron with 0, 1-vertices. *Journal of research of the National Bureau of Standards B*, *69*, 55–56.

[15] Gusfield, D. (1990). Very simple methods for all pairs network flow analysis. *SIAM Journal on Computing*, *19*, 143–155.

[16] Hochbaum, D. S., Lyu, C., & Ordóñez, F. (2014). Security routing games with multivehicle chinese postman problem. *Networks*, *64*, 181–191.

[17] Jain, M., Kardes, E., Kiekintveld, C., Ordoñez, F., & Tambe, M. (2010). Security games with arbitrary schedules: A branch and price approach. In *AAAI Conference on Artificial Intelligence*.

[18] Jain, M., Tsai, J., Pita, J., Kiekintveld, C., Rathi, S., Tambe, M., & Ordóñez, F. (2010). Software assistants for randomized patrol planning for the LAX airport police and the federal air marshal service. *Interfaces*, *40*, 267–290.

[19] Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Ordóñez, F., & Tambe, M. (2009). Computing optimal randomized resource allocations for massive security games. In *AAMAS* (pp. 689–696). volume 1.

[20] Li, Y., Conitzer, V., & Korzhyk, D. (2016). Catcher-evader games. *arXiv preprint arXiv:1602.01896*, .

[21] McClay, L. A., Lee, A. J., & Jacobson, S. (2010). Risk-based policies for airport security checkpoint screening. *Transportation Science*, *44*, 333–349.

[22] Nie, X., Batta, R., Drury, C. G., & Lin, L. (2009). Passenger grouping with risk levels in an airport security system. *European Journal of Operational Research*, *194*, 574–584.

[23] Padberg, M. W., & Rao, M. R. (1982). Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, *7*, 67–80.

[24] Papadaki, K., Alpern, S., Lidbetter, T., & Morton, A. (2016). Patrolling a border. *Operations Research*, *64*, 1256–1269.

[25] Paruchuri, P., Pearce, J. P., Marecki, J., Tambe, M., Ordóñez, F., & Kraus, S. (2008). Playing games for security: An efficient exact algorithm for solving bayesian stackelberg games. In *AAMAS* (pp. 895–902). volume 2.

[26] Shieh, E., An, B., Yang, R., Tambe, M., Baldwin, C., DiRenzo, J., Maule, B., & Meyer, G. (2012). Protect: A deployed game theoretic system to protect the ports of the United States. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1* (pp. 13–20).

[27] Thomas, M. U., & Nisgav, Y. (1976). An infiltration game with time dependent payoff. *Naval Research Logistics Quarterly*, *23*, 297–302.

[28] Wright, D., Liberatore, M., & Nydick, R. L. (2006). A survey of operations research models and applications in homeland security. *Interfaces*, *36*, 514–529.

[29] Xu, H. (2016). The mysteries of security games: Equilibrium computation becomes combinatorial algorithm design. In *Proceedings of the 2016 ACM Conference on Economics and Computation* (pp. 497–514). ACM.

[30] Yan, X., & Nie, X. (2016). Optimal placement of multiple types of detectors under a small vessel attack threat to port security. *Transportation Research Part E: Logistics and Transportation Review*, *93*.

[31] Yang, R., Ford, B., Tambe, M., & Lemieux, A. (2014). Adaptive resource allocation for wildlife protection against illegal poachers. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems* (pp. 453–460).

[32] Yang, R., Jiang, A. X., Tambe, M., & Ordonez, F. (2013). Scaling-up security games with boundedly rational adversaries: A cutting-plane approach. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

[33] Yin, Z., Jiang, A. X., Tambe, M., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., & Sullivan, J. P. (2012). Trusts: Scheduling randomized patrols for fare inspection in transit systems using game theory. *AI Magazine*, *33*, 59–72.

[34] Yin, Z., & Tambe, M. (2012). A unified method for handling discrete and continuous uncertainty in bayesian Stackelberg games. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2* (pp. 855–862).

Appendix A. Numerical Example of Algorithm 1

In this section we describe how Algorithm 1 works in a small instance with $m = 2$. Consider the example depicted in Figure A.8. Variables $\mathbf{z}$ are depicted in the edges and variables $\mathbf{c}$ are denoted next to the labels of the outposts. Variables $\mathbf{g}$ are represented in Table A.2.a.

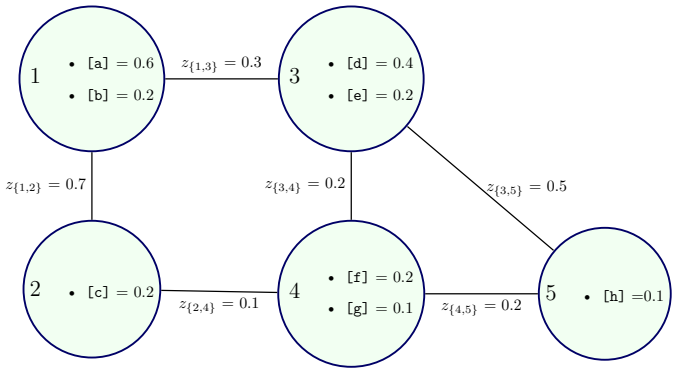


Figure A.8: Graph representing precincts and their outposts. Variables z and c are represented in edges and inside each node respectively.

Edge	Outpost	$\mathbf{g}$	Edge	Outpost	$\mathbf{g}$
(1,2)	a	0.4	(3.5)	d	0.3
	b	0.2		e	0.1
	c	0.1		h	0.1
(1,3)	a	0.2	(3,4)	d	0
	b	0		e	0.1
	d	0.1		f	0.1
	e	0		g	0
(2,4)	c	0.1	(4,5)	f	0.1
	f	0		g	0.1
	g	0		h	0

a)

Matching $\mathbf{y}$	$\lambda_{\mathbf{y}}$
$\mathbf{y}^1 : \{\{1,2\}, \{3,5\}\}$	0.5
$\mathbf{y}^2 : \{\{1,2\}, \{3,4\}\}$	0.2
$\mathbf{y}^3 : \{\{1,3\}, \{4,5\}\}$	0.2
$\mathbf{y}^4 : \{\{1,3\}, \{2,4\}\}$	0.1

b)

Table A.2: a) Values of variables $\mathbf{g}$ in the example. b) Decomposition of values $\mathbf{z}$ in pure matchings.

Given this data we retrieve a vector $\mathbf{x}$ satisfying (2.2), (2.15), (2.16) and (2.17). First we found a decomposition of vector $\mathbf{z}$ in a convex combination of pure matchings $\mathbf{y}$. It can be founded via the column generation procedure stated in Section 5.1. The vector $\mathbf{z}$ is decomposed in the four pure matchings and weights given in Table A.2.b.

The algorithm in the Step 1 creates a rectangle of height 1 and 6 columns, one for each edge. The height is in divided in 4 rows, one per each matching in Table A.2.b. The height of each row is equal to the value of $\lambda_{\mathbf{y}}$. For each edge, the algorithm labels with a 'NO' each sub-rectangle where the edge does not belong to the matching. For instance, the rectangle formed by the intersection of $\mathbf{y}^3$ and $\{1,2\}$ is labeled with a 'NO' because $\{1,2\}$ does not belong to $\mathbf{y}^3$. The output of this step is depicted in Figure A.9.a.

In Step 2, the algorithm fills each sub-rectangle with the values of $\mathbf{g}$. First, the algorithm fix an order of the outpost inside each edge Then, sub-columns of height g_{ej} are allocated in the column of the corresponding edge, such that the slot is not labeled with a 'NO'. This step is described in Figure A.9.b. It could be the case, as $g_{\{1,2\},b}$, that one value of g_{ej} lies between two different matchings.

In the final Step 3, the algorithm extends each rectangle line into a horizontal line crossing all columns. Some of them may coincide. Each horizontal area belongs to a matching and a set of locations covered by this matching. This procedure is depicted in A.9.c. For example, the first horizontal section from the bottom, returns a strategy in which precincts $\{1,2\}$ and $\{3,5\}$ are paired, and locations a and d are protected.

Appendix B. Proof of Lemma 1

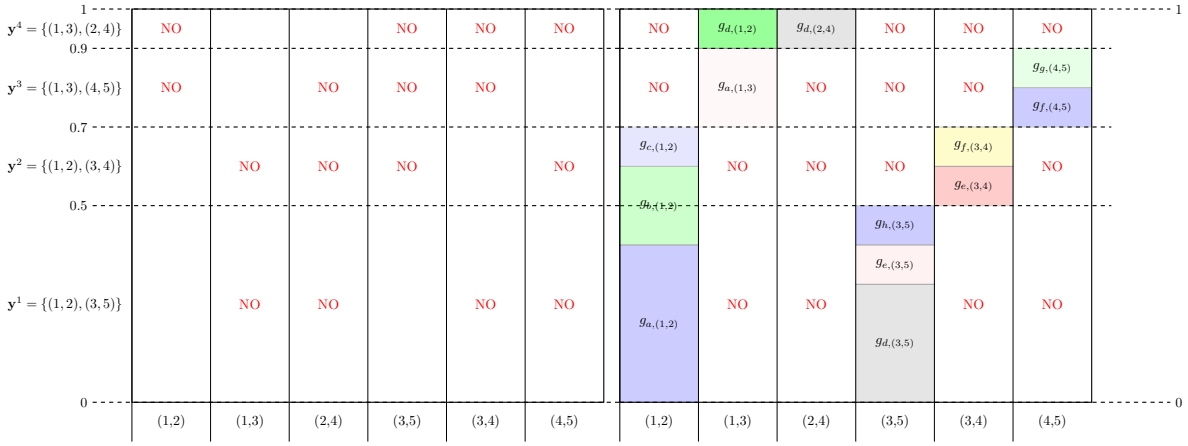
Lemma. *Given a set of matchings of size M_z and vectors $\mathbf{z}, \mathbf{c}, \mathbf{g}$, Algorithm 1 returns a vector x satisfying (2.2), (2.15), (2.16) and (2.17).*

Proof. We first show that all the $(\mathbf{y}, \mathbf{w})$ identified in Step 3 satisfy $(\mathbf{y}, \mathbf{w}) \in I$. First we have by construction that $\mathbf{y} \in \mathcal{M}_m$. In addition for each edge e used in $\mathbf{y}$ we identify a single target $j \in J_e$ so that $w_j = 1$. Therefore the vector $\mathbf{w}$ determines exactly m different targets. For any $U \subseteq V$, we have

$$\sum_{j \in \cup_{v \in U} J_v} w_j \leq \sum_{e\,:\,\{u_e, v_e\} \cap U \neq \emptyset} \; \sum_{j_e \in J_{u_e} \cup J_{v_e}} w_{j_e} = \sum_{e\,:\,\{u_e, v_e\} \cap U \neq \emptyset} y_e = \sum_{e \in E(U) \cup \delta(U)} y_e \ .$$

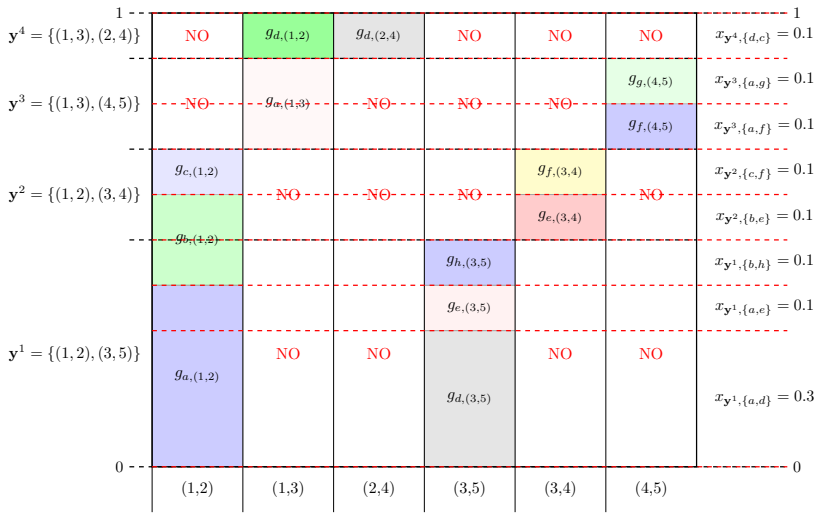
This proves that $(\mathbf{y}, \mathbf{w}) \in I$. By construction, the horizontal lines corresponding to defender strategies cover the box completely, since the entire area of the $1 \times |E|$ that is not blocked out is covered by some $g_{ej} > 0$. Therefore Step 3 can be performed at any height of the box corresponding to a rectangle limit line. This implies that the values $x_{(\mathbf{y},\mathbf{w})}$ constructed in Step 3 are non-negative and sum to one, therefore they form a probability distribution over $(\mathbf{y}, \mathbf{w}) \in I$, i.e. the vector $\mathbf{x}$ constructed satisfies (2.2).

Consider a target $j \in J$ and e such that $j \in J_e$ with $g_{ej} > 0$. We have from Step 2 in Algorithm 1 that the full amount of g_{ej} is assigned to some area in the column e intersected by matchings that use that column. That area can be partitioned into many strategies $(\mathbf{y}, \mathbf{w})$ but each of them has $y_e = 1$



a) Step 1

b) Step 2



c) Step 3

Figure A.9: Steps in Algorithm 1 for this Example.

and $w_j = 1$. Therefore we have that $g_{ej} = \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} w_j y_e$. We have then

$$c_j = \sum_{e:j\in J_e} g_{ej} = \sum_{e:j\in J_e} \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} w_j y_e = \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} w_j \sum_{e:j\in J_e} y_e = \sum_{(\mathbf{y},\mathbf{w})\in I} x_{(\mathbf{y},\mathbf{w})} w_j \ .$$

Where the last equality comes from (4.1). Then $\mathbf{x}$ satisfies expression (2.15). By the same argument, it is easy to check that condition (2.16) is also satisfied by $\mathbf{x}$. □

Appendix C. Proof of Lemma 2

Lemma. *Given* $(\mathbf{y},\mathbf{w}) \in I$ *then the following holds*

$$w_j = 1 \Rightarrow \sum_{e\in E:j\in J_e} y_e = 1$$
$$y_e = 1 \Rightarrow \sum_{j\in J_e} w_j = 1.$$

Proof. Since $(\mathbf{y},\mathbf{w}) \in I$ with $w_j = 1$, if we denote $v(j)$ the precinct that contains target j (i.e. $j \in J_{v(j)}$), we have

$$1 = w_j \leq \sum_{k\in J_{v(j)}} w_k \leq \sum_{e\in\delta(v(j))} y_e = \sum_{e\in E:j\in J_e} y_e \ ,$$

which proves (4.1). Here the first inequality comes from $w_k \in \{0,1\}$, the second from the definition of set I, and the last equality represents two equivalent ways of expressing the sum over the edges incident on $v(j)$. For the second implication, let $y_{e_1} = y_{e_2} = \cdots = y_{e_m} = 1$ be the set of arcs that are set to one in the matching y. Let $A(\mathbf{y})$ be the set of precincts v that are not paired by $y_{e_1} \dots y_{e_m}$ (i.e. $A(\mathbf{y}) = \{v \ : \ \delta(v) \cap \cup_{k=1}^{m} e_k = \emptyset\}$). Then, for any $v \in A(\mathbf{y})$ the definition of I implies $\sum_{k\in J_v} w_k \leq \sum_{e\in\delta(v)} y_e = 0$. Similarly, for any arc $e_k = (u_k, v_k)$ we have, from the definition of I, that $\sum_{j\in J_{e_k}} w_j \leq y_{e_k} + \sum_{e\in\delta(\{u_k,v_k\})} y_e = y_{e_k} = 1$. Here the sum in the second term is zero because $e \in \delta(\{u_k, v_k\})$ are not in matching y. With these inequalities we can separate

$$m = \sum_{j\in J} w_j = \sum_{v\in A(\mathbf{y})} \sum_{j\in J_v} w_j + \sum_{k=1}^{m} \sum_{j\in J_{e_k}} w_j \leq \sum_{k=1}^{m} y_{e_k} = m \ ,$$

which completes the proof since the m terms $\sum_{j\in J_{e_k}} w_j \leq 1$ and sum to m. □

Appendix D. Branch and Cut Implementation

The amount of blossom inequalities (2.23) can make this formulation untractable. Figure D.10 shows an example of a solution satisfying (2.21) and (2.22), but a fractional matching cannot be retrieved, that is, the solution does not belong to the matching polytope.

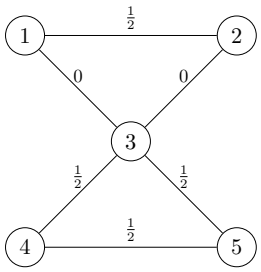


Figure D.10: Variables $\mathbf{z} \in [0,1]^{|E|}$ that satisfy (2.21) and (2.22) but violate (2.23) for $m = 2$.

Now we briefly explain how to separate this solution following the methodology in [23]. We begin by noting that the odd set inequalities (2.23) are equivalent to the following constraint:

$$\sum_{v \in U} s_v + \sum_{e \in \delta(U)} z_e \geq 1, \quad |U| \geq 3, \, |U| \text{ odd.} \tag{D.1}$$

This can be shown by adding slack variables s_v, $v \in V$ for every Constraint (2.22), summing Constraints (2.22) over a fixed subset of nodes U and noting that $\sum_{u \in U} \sum_{e \in \delta(u)} z_e = 2 \sum_{e \in E(U)} z_e + \sum_{e \in \delta(U)} z_e$.

Thus, a solution $(z^*, \{s_v^* = 1 - \sum_{e \in \delta(v)} z_e^*\}_{v \in V})$ violates an odd set inequality if there exists a subset of nodes U with odd cardinality such that $\sum_{v \in U} s_v^* + \sum_{e \in \delta(U)} z_e^* < 1$. To detect such condition is equivalent to detect a global min cut of odd cardinality in a suitable graph $G' = (V', E')$ where $V' = V \cup \{\mathcal{S}\}$, being $\mathcal{S}$ a dummy node and $E' = E \cup \{\{v, \mathcal{S}\} | v \in V\}$. The capacities are set z_e for each $e \in E$ and s_v for each of the form $\{v, \mathcal{S}\}$. The minimum cut of this network can be computed with the Gomory Hu algorithm and finding the odd cardinality set with minimum capacity. We utilise a simple implementation of the Gomory Hu algorithm given by [15]. If the capacity of the minimum odd cardinality set is smaller than one then the set of nodes U not containing $\mathcal{S}$ violates constraint (D.1). This procedure is described in Algorithm 2.

Figure D.11 shows an example of how the algorithm works cutting the infeasible solution exposed in Figure D.10. We represent the extended graph on the left and the min-cut set in dashed lines and show, on the right, how the Gomory-Hu tree computes this minimum cut. Note that in the example the min-cut odd set is formed by the set of nodes $\{3, 4, 5\}$ with cost 0 and the associated constraint is $z_{\{3,4\}} + z_{\{3,5\}} + z_{\{4,5\}} \leq 1$ which cuts the solution in Figure D.10.

We implement this cuts by using Lazy Constraints Callbacks in CPLEX 12.7. In this rputine, each time that an integer solution is found, 2 detects if this solution satisfies all the set of Constraints (2.23). If this is the case, an optimal solution is found. Otherwise, the violated constraint is added to the Branch and Bound scheme and the search continues.

Algorithm 2 Min odd-cut set

Graph $G = (V, E)$, $z \in \mathbb{R}_+^{|E|}$

Create a graph $G' = (V', E')$ where $V' = V \cup \{\mathcal{S}\}$ and $E' = E \cup \{\{v, \mathcal{S}\} | v \in V\}$, and capacities $\bar{r}_{e'} = z_{e'}$ for edges $e' \in E$ and $\bar{r}_{e'} = s_v$ for edges $e' = (v, \mathcal{S}) \in E' \backslash E$.

Compute a Gomory - Hu tree $G_T = (V_T, E_T)$ in the graph G' and set $r = +\infty$

for $e \in E_T$ **do**

  Calculate the cut U induced by deleting e in E_T.

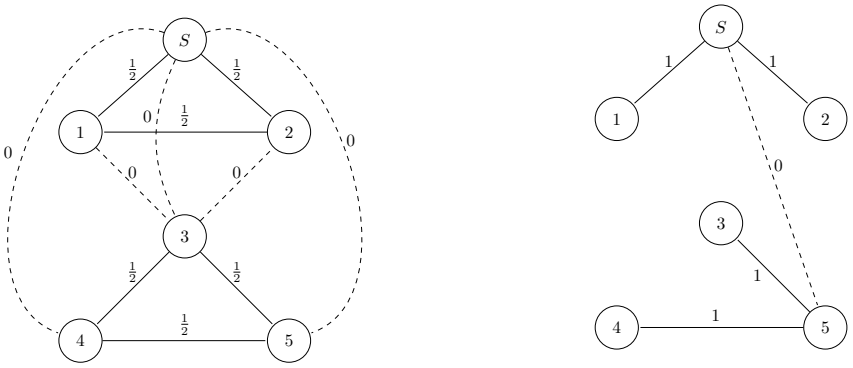
  if $|U|$ is odd and $r_e < r$ **then**

    $X \leftarrow U$ and $r \leftarrow r_e$

  end if

end for

return X the odd minimum cut-set and r its capacity.



a) Extended graph of solution in Fig. D.10

b) Gomory-Hu Tree

Figure D.11: Representation of the algorithm which detects the Odd min cut set. The dashed edges shows the min cut of this graph.