Approximation Algorithm for Security Games with Costly Resources

Sayan Bhattacharya, Vincent Conitzer, and Kamesh Munagala

Department of Computer Science, Duke University
Durham, NC 27708, USA
{bsayan,conitzer,kamesh}@cs.duke.edu

Abstract. In recent years, algorithms for computing game-theoretic solutions have been developed for real-world security domains. These games are between a defender, who must allocate her resources to defend potential targets, and an attacker, who chooses a target to attack. Existing work has assumed the set of defender's resources to be fixed. This assumption precludes the effective use of approximation algorithms, since a slight change in the defender's allocation strategy can result in a massive change in her utility. In contrast, we consider a model where resources are obtained at a cost, initiating the study of the following optimization problem: Minimize the total cost of the purchased resources, given that every target has to be defended with at least a certain probability. We give an efficient logarithmic approximation algorithm for this problem.

1 Introduction

Taking a game as input and computing a solution of it is one of the core problems of algorithmic game theory. To be more precise, it is a collection of problems, one for each combination of a representation scheme and a solution concept. Perhaps the best-known example is the problem of computing a Nash equilibrium of a normal-form game, which is now known to be PPAD-complete for two-player games [2, 3] and FIXP-complete for games with three or more players [5].

In contrast, this paper deals with an alternative solution concept, corresponding to a "Stackelberg" model. In this model, there are two players, a "leader" and a "follower". The leader first commits to a mixed strategy, and the follower responds only after observing the commitment. Recently, algorithms for computing optimal leader strategies have been developed for various real-world security domains, including US airports [11, 12], the US Federal Air Marshals [13], and the US Coast Guard [1].

Motivated by such applications, Kiekintveld *et al.* [8] proposed a general model of security games. The players are a *defender* and an *attacker*. There is a set of *targets* that the attacker may want to attack (in the Federal Air Marshal example, these would be individual flights), and the defender has a set of *resources* (Federal Air Marshals). The defender can assign each resource to a *schedule*, which consists of a subset of the targets (for example, a tour of multiple flights that a single Federal Air Marshal can take). In general, not every resource

can be assigned to every schedule. If a resource is assigned to a schedule, then it defends all the targets contained in that schedule.

A pure (resp., mixed) strategy for the defender is a deterministic (resp., randomized) assignment of the resources to schedules. The typical motivation given for the Stackelberg model, where the defender (the leader) commits to a mixed strategy and the attacker (the follower) subsequently best-responds, is as follows. Every day, the defender draws an assignment from her distribution. Over the course of time, the attacker observes the realized assignments and eventually learns the probabilities with which each target is defended on any given day. Then, the attacker decides to attack a single target that maximizes her expected utility. The utilities of both players usually depend on: (a) the target that is attacked, and (b) whether or not that target is defended on the day of the attack. In the existing literature on this topic, the typical problem is to find the optimal mixed strategy for the defender to commit to, that is, the one that will maximize the defender's expected utility.

In this paper, we explore the setting where the resources can be purchased by the defender at a cost. To be more specific, we consider the following problem. Some resources are to be (deterministically) purchased in advance. Next, they are to be randomly assigned to schedules, ensuring that each target is defended with at least a certain probability. The objective is to minimize the total cost of the purchased resources.

For example, suppose that the Federal Air Marshal Service can hire a set of Marshals on a contractual basis for one year, thereby incurring a certain cost. During each day of the following year, the hired Marshals are (randomly) assigned to the schedules, according to the mixed strategy chosen. It is important to emphasize that we require a set of resources to be purchased *deterministically*, after which these resources can be randomly assigned to schedules. It is impractical to randomize the resource supply itself each day, since, for example, security personnel cannot be hired as day laborers.

Existing literature [8, 9] on security games assumes a fixed set of resources. In contrast, our model assumes that the resources are available in unlimited supply, but they can be purchased at a cost. There is a connection between these two settings: Given a fixed set of resources, there is an efficient algorithm to compute the optimal commitment strategy for a defender *if and only if* we can decide in polynomial time whether each target can be defended with a certain probability. The proof of this claim appears in the full version of our paper.

Our Results and Techniques. In Section 2, we formally state our problem and show that it is a generalization of Set Cover, and hence the problem is unlikely to admit an approximation ratio better than $O(\log |\mathcal{T}|)$, where $|\mathcal{T}|$ is the number of targets [6]. Section 3 gives an $O(\log |\mathcal{T}| + \log |\mathcal{S}|)$ approximation algorithm for this problem, where $|\mathcal{S}|$ is the number of schedules. The algorithm partitions the set of targets into two subsets, depending on whether their required coverage probabilities are *small* or *big*, and separately deals with these two subsets.

In Section 3.1, we show that the subproblem of defending the targets with big probabilities admits an LP relaxation (LP (3)), and it can be converted to the

LP relaxation for Set Cover (LP (6)) while losing at most a constant factor in the objective value. Thus, a greedy algorithm (Figure 1) gives a good approximation for this case. Unfortunately, this analysis cannot be extended to defend the targets with small probabilities, because the gap between the optimal objective value of LP (3) and the cost of the optimal solution can become arbitrarily large.

Section 3.2 considers the remaining subproblem—to defend the targets with small probabilities. For this, we present an exponentially sized covering *integer* program (IP (9)) that bounds the cost (Lemma 3) of the optimal solution. Interestingly, the LP relaxation of IP (9) can again be far away from the optimal solution, and thus, it is necessary to impose the integrality constraints.

IP (9) has the additional property that all of its constraint data are integral. Under such circumstances, a simple greedy heuristic (see Dobson [4]) gives a logarithmic approximation to the integer optimum (Lemma 4). The idea is to treat every variable as a set that *covers* part of the constraints, and during each iteration, increment the variable that gives maximum coverage per unit cost. However, IP (9) has exponentially many variables, and hence, we cannot directly apply Dobson's algorithm to solve it in polynomial time. Instead, we show that the subroutine of selecting the variable with maximum coverage per unit cost is exactly equivalent to maximizing a submodular function subject to a budget constraint. Hence, we can use another greedy algorithm [10] to implement this subroutine upto a constant factor approximation. Accordingly, we solve IP (9), and its outcome determines the resources we purchase and the way in which we randomly assign them to the schedules (Theorem 2).

Remark. All the missing proofs appear in the full version of this paper.

2 Notations and Preliminaries

There are a set of targets $\mathcal{T}$, a set of schedules $\mathcal{S}$, and a set of resource-types Θ. There is an *unlimited supply* of resources of each type. Any resource of type $\theta \in \Theta$ has cost $c(\theta)$. In addition, there is a subset $S(\theta) \subseteq \mathcal{S}$ such that any resource of type θ can be *assigned* to at most one schedule in $S(\theta)$. The type of a resource r is denoted by $\theta_r \in \Theta$. Whenever some resource is assigned to schedule $s \in \mathcal{S}$, it *defends* all targets in the subset $T(s) \subseteq \mathcal{T}$. Furthermore, each target $t \in \mathcal{T}$ has a *threshold requirement* $0 \leq q_t \leq 1$. In the Security Game problem, we want to (deterministically) purchase some resources, and randomly assign them to schedules so that every target $t \in \mathcal{T}$ is defended with probability at least q_t. We want to minimize the sum of the costs of the purchased resources.[1]

Consider an example. There are 4 targets, 3 schedules, and 2 different types of resources. Target t_1 (resp. t_4) needs to be defended with probability 2/3 (resp.

[1] Note that the unlimited availability of resources guarantees the existence of a feasible solution satisfying all the threshold requirements: simply purchase a sufficient number of resources of each type. This assertion holds provided each target can be defended by some schedule, and for each schedule, there is some resource that can be assigned to it. We will make these assumptions without any loss of generality.

1/3), whereas each of the targets in $\{t_2, t_3\}$ has a threshold requirement of 1. Any resource of type θ_1 costs 2, and any resource of type θ_2 costs 3. A resource of type θ_1 (resp. θ_2) can be assigned to at most one schedule in the set $\{s_1, s_2\}$ (resp. $\{s_2, s_3\}$). Finally, whenever it has some resource assigned to it, schedule s_1 defends target t_1; schedule s_2 defends both the targets t_2, t_3; and schedule s_3 defends both the targets t_3, t_4. In terms of notations, we have

$$\begin{aligned} \mathcal{T} = \{t_1, t_2, t_3, t_4\},\ \mathcal{S} = \{s_1, s_2, s_3\},\ \Theta = \{\theta_1, \theta_2\} \\ q_{t_1} = 2/3,\ q_{t_2} = q_{t_3} = 1,\ q_{t_4} = 1/3 \\ c(\theta_1) = 2,\ c(\theta_2) = 3 \\ S(\theta_1) = \{s_1, s_2\},\ S(\theta_2) = \{s_2, s_3\} \\ T(s_1) = \{t_1\},\ T(s_2) = \{t_2, t_3\},\ T(s_3) = \{t_3, t_4\} \end{aligned}$$

The optimal solution will purchase one resource r_1 of type θ_1 and one resource r_2 of type θ_2 so that $\theta_{r_1} = \theta_1,\ \theta_{r_2} = \theta_2$; thereby incurring a total cost of $2+3=5$. Next, with probability $2/3$, it will *simultaneously* assign resource r_1 to schedule s_1 and resource r_2 to schedule s_2; and with the remaining probability $1-2/3 = 1/3$, it will *simultaneously* assign resource r_1 to schedule s_2 and resource r_2 to schedule s_3. It is important to note how the optimal solution correlates the random assignments of the resources to schedules.

The Security-Game problem is a generalization of Set-Cover. Consider an instance of the Security-Game problem where the threshold requirements of all the targets are equal to 1, and there is only one resource-type. In this case, the task of finding the optimal solution is equivalent to finding a minimum cardinality subset of schedules to defend all the targets, which is exactly the Set-Cover problem. As a consequence, the Security-Game problem is NP-hard and unless NP has slightly superpolynomial time algorithms, we cannot even approximate it to a factor better than $O(\log |\mathcal{T}|)$ [6]. In the next section, we give an $O(\log |\mathcal{T}| + \log |\mathcal{S}|)$ approximation algorithm.

3 Approximation Algorithm

First, we partition the set of targets into two groups depending on their threshold requirements. Define

$$\mathcal{T}_{big} = \{t \in \mathcal{T} \ :\ 1/e < q_t \leq 1\} \tag{1}$$

$$\mathcal{T}_{small} = \{t \in \mathcal{T} \ :\ q_t \leq 1/e\} \tag{2}$$

We deal with the two subsets separately. In Section 3.1, we consider the subproblem where we need to defend *only* the subset $\mathcal{T}_{big}$ of targets, and give an $O(\log |\mathcal{T}|)$ approximation algorithm for this task. On the other hand, Section 3.2 deals with the subproblem where we have to defend *only* the targets $t \in \mathcal{T}_{small}$. For this task, we present an $O(\log |\mathcal{T}| + \log |\mathcal{S}|)$ approximation algorithm. Finally, we take the union of the two solutions, and this results in an $O(\log |\mathcal{T}| + \log |\mathcal{S}|)$ approximation for the Security-Game problem.

3.1 Targets with Big Threshold Requirements

Let OPT_{big} denote the minimum-cost solution that only defends the targets in the subset $\mathcal{T}_{big} \subseteq \mathcal{T}$ according to their threshold requirements, and ignores the remaining targets in $\mathcal{T}_{small} = \mathcal{T} \setminus \mathcal{T}_{big}$. We now derive an LP-relaxation.

$$\min \quad \sum_{\theta \in \Theta} c(\theta) \sum_{s \in S(\theta)} w(\theta, s) \tag{3}$$

$$\text{s.t.} \quad \sum_{\theta \in \Theta} \sum_{s \in S(\theta):\, t \in T(s)} w(\theta, s) \geq q_t, \quad \forall t \in \mathcal{T}_{big} \tag{4}$$

$$w(\theta, s) \geq 0 \qquad , \quad \forall \theta \in \Theta, s \in S(\theta) \tag{5}$$

Let $\mathcal{R}^*$ be the set of resources purchased by OPT_{big}. For all $r \in \mathcal{R}^*, s \in S(\theta_r)$, let y^*_{rs} be the probability that OPT_{big} assigns resource r to schedule s. Recall that the type of a resource r is denoted by θ_r. Define

$$w^*(\theta, s) = \sum_{r \in \mathcal{R}:\, \theta_r = \theta} y^*_{rs}$$

It is easy to verify that the $w^*(\theta, s)$ values are a feasible solution to LP (3). Constraint (4) holds since OPT_{big} defends every $t \in \mathcal{T}_{big}$ with probability at least q_t and by the union bound, the left hand side of the constraint is an overestimate of the probability with which target t is defended. The relaxation assumes that the resources can be purchased partially and hence, the objective value is at most the total cost incurred by OPT_{big}. This leads us to Lemma 1.

Lemma 1. *LP (3) gives a lower bound on the total cost incurred by OPT_{big}.*

Consider the following linear program (6). It replaces the right hand of Constraint (4) by 1. Recall that all targets in the subset $\mathcal{T}_{big} \subseteq \mathcal{T}$ have $q_t \geq 1/e$. Therefore, if we solve LP (3) optimally and multiply every $w(\theta, s)$ by e, then we get a feasible solution to LP (6). However, the objective value also increases by a factor of e. Combining this observation with Lemma 1, we obtain Fact 1.

$$\min \quad \sum_{\theta \in \Theta} c(\theta) \sum_{s \in S(\theta)} w(\theta, s) \tag{6}$$

$$\text{s.t.} \quad \sum_{\theta \in \Theta} \sum_{s \in S(\theta):\, t \in T(s)} w(\theta, s) \geq 1, \quad \forall t \in \mathcal{T}_{big} \tag{7}$$

$$w(\theta, s) \geq 0 \qquad , \quad \forall \theta \in \Theta, s \in S(\theta) \tag{8}$$

Fact 1 *The optimal objective value of LP (6) is at most $O(1)$ (specifically, e) times the cost incurred by OPT_{big}.*

We note that LP (6) is an LP-relaxation for the Set-Cover problem, where the targets $t \in \mathcal{T}_{big}$ behave like elements that have to be covered, and each pair (θ, s) acts like a set $T(s) \cap \mathcal{T}_{big}$ having a cost $c(\theta)$. Hence the greedy algorithm described in Figure 1 gives a $O(\log |\mathcal{T}|)$ approximation [7] to LP (6). Hence, Fact 1 implies the following Theorem 1.

Initialize $D \leftarrow \emptyset,\ F \leftarrow \emptyset$.
While $D \neq \mathcal{T}_{big}$ do
    Find an ordered pair $(\theta', s') \in \arg\max_{\theta \in \Theta, s \in S(\theta)} |(T(s) \cap \mathcal{T}_{big}) \setminus D|/c(\theta)$
    $F \leftarrow F \cup \{(\theta', s')\}$
    $D \leftarrow D \cup (T(s') \cap \mathcal{T}_{big})$
For all $(\theta, s) \in F$ do
    Buy a resource of type θ, and *deterministically* assign it to schedule s.

Fig. 1. Greedy algorithm for targets with big threshold requirements. It defends every target $t \in \mathcal{T}_{big}$ with probability 1.

Theorem 1. *The greedy algorithm described in Figure (1) gives an* $O(\log |\mathcal{T}|)$ *approximation to* OPT_{big}.

3.2 Targets with Small Threshold Requirements

Let SG_{small} be the problem where we must defend each target $t \in \mathcal{T}_{small}$ with probability at least q_t, but we are free to ignore the remaining targets in $\mathcal{T}_{big}$. A *solution* to the SG_{small} problem purchases some resources and randomly assigns them to schedules so that each target $t \in \mathcal{T}_{small}$ is defended with probability q_t.

Definition 1. *Given any solution to the* SG_{small} *problem, every purchased resource* r *can be associated with an* assignment vector. *It is a vector with* $|\mathcal{S}|$ *components, where the value of component* s *equals* y_{rs} *if* $s \in S(\theta_r)$ *and is zero otherwise. Here,* y_{rs} *is the probability that resource* r *is assigned to schedule* s.

Lemma 2 shows that we can restrict our attention to a subset of feasible solutions to the SG_{small} problem. Specifically, we focus on those solutions where the values of all components of the assignment vectors come from a discrete set.

Lemma 2. *Let* Opt *be the minimum-cost solution to the* SG_{small} *problem. There exists a solution* Discrete-Opt *to the* SG_{small} *problem such that:*

1. *The cost incurred by* Discrete-Opt *is at most* 2 *times the cost of* Opt.
2. *For every resource purchased by* Discrete-Opt, *all the components of the corresponding assignment vector are integral multiples of* $1/|\mathcal{S}|^2$.

Fix any target $t \in \mathcal{T}_{small}$ with $0 < q_t < 1/|\mathcal{S}|^2$. The solution Discrete-Opt defends this target according to its threshold requirement. Thus, Discrete-Opt purchases some resource of type θ, and assigns it with *non-zero* probability to some schedule $s \in S(\theta)$ such that $t \in T(s)$. However, the probability of assigning the resource to schedule s is an integral multiple of $1/|\mathcal{S}|^2$ (Lemma 2), and hence the target t is defended with probability at least $1/|\mathcal{S}|^2$.

Corollary 1. *The solution* Discrete-Opt *defends each target* $t \in \mathcal{T}_{small}$ *with probability at least* $\min(q_t, 1/|\mathcal{S}|^2)$.

Let $\mathcal{P}$ denote the set of all assignment vectors that have been discretized according to Lemma 2. More formally, the set $\mathcal{P}$ consists of all possible $|\mathcal{S}|$-tuples where the value of each component is an integral multiple of $1/|\mathcal{S}|^2$, and the sum of the values of all the components is at most 1. For all $\boldsymbol{p} \in \mathcal{P}, s \in \mathcal{S}$, let $\boldsymbol{p}(s)$ denote the component of the assignment vector $\boldsymbol{p}$ corresponding to schedule s.

Suppose that a resource r cannot be assigned to some schedule s, that is, $s \notin S(\theta_r)$, and a valid solution assigns the resource r to different schedules with probabilities that are given by the components of the vector $\boldsymbol{p} \in \mathcal{P}$. In this case, we must have $\boldsymbol{p}(s) = 0$. Definition 2 formalizes this concept.

Definition 2. *An assignment vector $\boldsymbol{p} \in \mathcal{P}$ is* feasible *for a resource-type $\theta \in \Theta$ if $\boldsymbol{p}(s) = 0$ for all schedules $s \in \mathcal{S} \setminus S(\theta)$. Define $\mathcal{P}_\theta$ to be the set of all feasible assignment vectors for resource type $\theta \in \Theta$.*

Now we present an *Integer* Program to bound the cost of DISCRETE-OPT.

$$\min \quad \sum_{\theta \in \Theta,\, \boldsymbol{p} \in \mathcal{P}_\theta} c(\theta) x(\theta, \boldsymbol{p}) \tag{9}$$

$$\text{s.t.} \quad \sum_{\theta \in \Theta,\, \boldsymbol{p} \in \mathcal{P}_\theta} \; \sum_{s:\, t \in T(s)} \eta(\boldsymbol{p}, s)\, x(\theta, \boldsymbol{p}) \geq \lambda_t, \quad \forall t \in \mathcal{T}_{small} \tag{10}$$

$$x(\theta, \boldsymbol{p}) \in \mathbb{N} \qquad , \quad \forall \theta \in \Theta,\, \boldsymbol{p} \in \mathcal{P}_\theta \tag{11}$$

IP (9) introduces some new notation. In Constraint (11), the set of all non-negative integers is denoted by $\mathbb{N}$. In Constraint (10), we have

$$\lambda_t = \lceil e \times q_t \times |\mathcal{S}|^2 \rceil \qquad \text{for all } t \in \mathcal{T}_{small} \tag{12}$$

$$\eta(\boldsymbol{p}, s) = \boldsymbol{p}(s) \times |\mathcal{S}|^2 \qquad \text{for all } \boldsymbol{p} \in \mathcal{P},\, s \in \mathcal{S} \tag{13}$$

By definition, each $\boldsymbol{p}(s) \in [0, 1]$ is an integral multiple of $1/|\mathcal{S}|^2$, and $q_t \in [0, 1/e]$ for all $t \in \mathcal{T}_{small}$. These observations lead to the the following fact.

Fact 2 *IP (9) is a covering integer program where all the coefficients in the constraint data, that is, all the values of $\eta(\boldsymbol{p}, s)$ and λ_t, are integers lying between 0 and $|\mathcal{S}|^2$.*

Lemma 3. *The optimal objective value of the Integer Program (9) is at most 4 times the cost incurred by* DISCRETE-OPT.

We now describe some intuitions behind IP (9). The variable $x(\theta, \boldsymbol{p})$ denotes the number of purchased resources that satisfy *both* of the following conditions.

1. The resource is of type $\theta \in \Theta$, *and*
2. For all $s \in \mathcal{S}$, the probability that the resource is assigned to schedule s is given by $\boldsymbol{p}(s)$.

Each resource of type θ costs $c(\theta)$. Therefore, summing over all possible resource types and feasible assignment vectors, we see that the total cost is given by the objective function. We now proceed towards verifying Constraint (10). Applying

union-bound, we can show that the left hand side of Constraint (10) is at least $|\mathcal{S}|^2$ times the probability of defending target t. Glossing over some of the details, the constraint holds since, the right hand side is roughly equal to $|\mathcal{S}|^2$ times the probability of defending target t.[2]

1. Let $\boldsymbol{\delta}$ denote a vector with $|\mathcal{T}_{small}|$ components, where $\boldsymbol{\delta}(t)$ gives the value of the component corresponding to target $t \in \mathcal{T}_{small}$.
2. FOR ALL $t \in \mathcal{T}_{small}$: Initialize $\boldsymbol{\delta}(t) \leftarrow \lambda_t$.
3. FOR ALL $\theta \in \Theta$, $\boldsymbol{p} \in \mathcal{P}_\theta$: Initialize $X(\theta, \boldsymbol{p}) \leftarrow 0$.
4. WHILE $\boldsymbol{\delta} \neq 0$ DO
5. FOR ALL $\boldsymbol{p} \in \mathcal{P}$, and $t \in \mathcal{T}_{small}$
$\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta}, t) \leftarrow \min\left(\boldsymbol{\delta}(t), \sum_{s:\, t \in T(s)} \boldsymbol{p}(s) \times |\mathcal{S}|^2\right)$.
6. FOR ALL $\boldsymbol{p} \in \mathcal{P}$: $\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta}) \leftarrow \sum_{t \in \mathcal{T}_{small}} \Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta}, t)$.
7. Find some $(\theta, \boldsymbol{p}) \in \arg\max_{\theta \in \Theta,\, \boldsymbol{p} \in \mathcal{P}_\theta} \left\{\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta})/c(\theta)\right\}$.
8. $X(\theta, \boldsymbol{p}) \leftarrow X(\theta, \boldsymbol{p}) + 1$.
9. FOR ALL $t \in \mathcal{T}_{small}$: $\boldsymbol{\delta}(t) \leftarrow \boldsymbol{\delta}(t) - \Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta}, t)$.
10. Return the $X(\theta, \boldsymbol{p})$ values for all $\theta \in \Theta$, $\boldsymbol{p} \in\in \mathcal{P}_\theta$.

Fig. 2. Dobson's Algorithm applied to LP (9)

If a covering *integer* program has a constraint matrix with integral entries, then a simple greedy heuristic returns a logarithmic approximation to the *integral* optimum (Dobson [4]). Fact 2 tells us that we can apply Dobson's heuristic to IP (9). A simple intuition behind the algorithm (Figure 2) comes from an alternate way of viewing the problem: Each target t has to be *covered* by a threshold amount λ_t. The total *coverage* required is $\sum_{t \in \mathcal{T}_{small}} \lambda_t$. This coverage can be achieved by incrementing the *columns* $\{(\theta, \boldsymbol{p})\}$, where each column $(\theta, \boldsymbol{p})$ corresponds to the variable $x(\theta, \boldsymbol{p})$. We want to increment the columns so that the required coverage is attained at minimum cost.

At the beginning of a typical iteration of the WHILE loop (Steps 4–9), the value of $\boldsymbol{\delta}(t)$ equals the *remaining coverage* required for target t before we can attain its threshold λ_t. If we increment a column $(\theta, \boldsymbol{p})$ by 1, then the cost of our solution will increase by $c(\theta)$, and at the same time, the coverage of target t will increase (Step 5) by the amount $\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta}, t)$. Hence, the increase in total coverage of all the targets (Step 6) will be equal to $\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta})$. Let us term this quantity $\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta})$ as *marginal coverage*. The algorithm myopically selects the column that has the maximum marginal coverage to cost ratio (Step 7), and increments that column by 1 (Step 8). The remaining coverage required for all the targets are adjusted accordingly (Step 9). The WHILE loop terminates

[2] To be more precise, the RHS is equal to $\lceil e \times q_t \times |\mathcal{S}|^2 \rceil$. While converting the IP solution to a feasible (random) assignment of resources to schedules, the final algorithm (Figure 3) looses a factor of e in the probability of defending any target $t \in \mathcal{T}_{small}$.

(Step 4) when $\boldsymbol{\delta} = 0$, that is, when all the targets in $\mathcal{T}_{small}$ have been covered up to their corresponding thresholds.

Note that IP (9) contains exponentially many variables $x(\theta, \boldsymbol{p})$. This follows from the fact that the set $\mathcal{P}$ of possible assignment vectors is exponential in size. Hence, we have to prove that Dobson's algorithm can be implemented in polynomial time. We also need to establish a bound on the approximation ratio.

Lemma 4. *Dobson's algorithm (Figure 2) can be used to solve the Integer Program (9). Although IP (9) contains exponentially many variables, an approximate version of Dobson's algorithm can be implemented in polynomial time. It returns a feasible solution to IP (9), where each variable $x(\theta, \boldsymbol{p})$ is assigned a nonnegative integral value $X(\theta, \boldsymbol{p})$. The solution satisfies two properties.*

1. *The objective value of the solution is at most $O(\log |\mathcal{T}| + \log |\mathcal{S}|)$ times the optimal objective value of IP (9).*
2. *The number of variables taking nonzero values are polynomially bounded.*

Proof (Sketch). The approximation ratio of Dobson's algorithm [4] grows logarithmically with the maximum column sum of the coefficient matrix. Recall that (Fact 2) each $\eta(\boldsymbol{p}, s)$ is an integer lying between 0 and $|\mathcal{S}|^2$, and the number of constraints in IP (9) is at most $|\mathcal{T}|$. Thus, in case of IP (9), the maximum column sum is upper bounded by $|\mathcal{S}|^2 \times |\mathcal{T}|$. Hence the approximation ratio is given by $O(\log(|\mathcal{S}|^2\,|\mathcal{T}|)) = O(\log |\mathcal{S}| + \log |\mathcal{T}|)$. Next, we will show that an approximate version of Dobson's algorithm can be implemented in polynomial time, and asymptotically, it gives the same approximation ratio of $O(\log |\mathcal{S}| + \log |\mathcal{T}|)$.

It is sufficient to discuss the implementations of Step 3, the WHILE loop (Steps 4–9) and Step 10. We implement Step 3 by implicitly assuming that all the $X(\theta, \boldsymbol{p})$ values have been initialized to zero. During the course of the algorithm, we keep track of only those $X(\theta, \boldsymbol{p})$ values that have been incremented at least once. Since each λ_t is an integer lying between 0 and $|\mathcal{S}|^2$ (Fact 2), the total coverage required of all targets is at most $|\mathcal{T}| \times |\mathcal{S}|^2$. Every iteration of the WHILE loop (Steps 4–9) contributes at least 1 towards this total coverage, and it increments exactly one $X(\theta, \boldsymbol{p})$. Therefore, at the termination of the WHILE loop, the number of nonzero $X(\theta, \boldsymbol{p})$ values is upper bounded by the polynomial $|\mathcal{T}| \times |\mathcal{S}|^2$. In Step 10, the algorithm returns only these nonzero $X(\theta, \boldsymbol{p})$ values, and all other variables are implicitly set to zero.

Regarding the WHILE loop (Steps 4–9), note that the marginal coverage $\Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta})$, as a function of the assignment vector $\boldsymbol{p}$, is monotone and submodular. To be more precise, fix some resource type θ. Next, take any two assignment vectors $\boldsymbol{p}, \boldsymbol{p}' \in \mathcal{P}_\theta$ such $\boldsymbol{p}$ is dominated by $\boldsymbol{p}'$, that is, $\boldsymbol{p}(s) \leq \boldsymbol{p}'(s)$ for all $s \in \mathcal{S}$. Furthermore, suppose that $\sum_{s \in \mathcal{S}} \boldsymbol{p}'(s) < 1$. Fix some schedule $s^* \in S(\theta)$ and consider two new assignment vectors $\boldsymbol{p}_1, \boldsymbol{p}'_1 \in \mathcal{P}_\theta$ with the following properties. For all $s \in \mathcal{S} \setminus \{s^*\}$, we have $\boldsymbol{p}_1(s) = \boldsymbol{p}(s)$, and $\boldsymbol{p}'_1(s) = \boldsymbol{p}'(s)$. We also have $\boldsymbol{p}_1(s^*) = \boldsymbol{p}(s^*) + 1/|\mathcal{S}|^2$, and $\boldsymbol{p}'_1(s^*) = \boldsymbol{p}'(s^*) + 1/|\mathcal{S}|^2$. Now, submodularity of marginal coverage means that the next inequality will always be satisfied.

$$\Delta_{Cov}(\boldsymbol{p}'_1, \boldsymbol{\delta}) - \Delta_{Cov}(\boldsymbol{p}', \boldsymbol{\delta}) \leq \Delta_{Cov}(\boldsymbol{p}_1, \boldsymbol{\delta}) - \Delta_{Cov}(\boldsymbol{p}, \boldsymbol{\delta})$$

We can exploit this submodularity condition as follows. While implementing Step 7, suppose we have correctly guessed the resource type θ that maximizes the marginal coverage to cost ratio.[3] All we need to do is to find an assignment vector $\boldsymbol{p} \in \mathcal{P}_\theta$ with maximum marginal coverage, subject to the constraints that each component of the vector $\boldsymbol{p}$ is an integral multiple of $1/|\mathcal{S}|^2$, and the sum of all the components is at most *one* (Lemma 2). This is equivalent to maximizing a monotone submodular function subject to a budget constraint [10]. A simple greedy algorithm is known to give a $(1 - 1/e)$ approximation for this problem.

To summarize, in polynomial time we can obtain a column $(\theta, \boldsymbol{p})$ that gives a constant approximation to the optimal ratio of marginal coverage to cost. Going through Dobson's proof [4], it is easy to verify the following statement. Even if we implement Step 7 in this approximate fashion, the algorithm will have the same asymptotic approximation guarantee. This concludes the proof. □

We are now ready to present our algorithm for the SG_{small} problem (Figure 3). First, we solve IP (9) according to Lemma 4. Let B^* denote the set of columns of IP (9) where the corresponding variable is set to some nonzero value, that is, $B^* = \{(\theta, \boldsymbol{p}) : X(\theta, \boldsymbol{p}) \neq 0\}$. For each $(\theta, \boldsymbol{p}) \in B^*$, we purchase $X(\theta, \boldsymbol{p})$ resources of type θ and tag them with the assignment vector $\boldsymbol{p}$. Finally, each purchased resource is randomly assigned to some schedule according to its assignment vector; and this process occurs *independently* of all other resources.

Solve IP (9) according to Lemma 4.
Define $\mathcal{P}^*_\theta = \{\boldsymbol{p} \in \mathcal{P}_\theta : X(\theta, \boldsymbol{p}) \neq 0\}$ for all $\theta \in \Theta$.
Let $\mathcal{R}_\theta$ be the set of resources of type θ that will be purchased.
Let $\mathcal{R} = \bigcup_{\theta \in \Theta} \mathcal{R}_\theta$ be the set of *all* resources that will be purchased.
Let $\mathcal{R}_{\theta,\boldsymbol{p}} \subseteq \mathcal{R}_\theta$ denote the resources in $\mathcal{R}_\theta$ whose
assignment probabilities are specified by $\boldsymbol{p} \in \mathcal{P}^*_\theta$.
For all resource-types $\theta \in \Theta$
    $|\mathcal{R}_\theta| = \sum_{\boldsymbol{p} \in \mathcal{P}^*_\theta} X(\theta, \boldsymbol{p})$.
    For all $\boldsymbol{p} \in \mathcal{P}^*_\theta$: $|\mathcal{R}_{\theta,\boldsymbol{p}}| = X(\theta, \boldsymbol{p})$.
Randomly assign each resource $r \in \mathcal{R}_{\theta,\boldsymbol{p}}$ to schedules, according to the
assignment probabilities specified by $\boldsymbol{p}$, independently of all other resources.

Fig. 3. Approximation Algorithm for the SG_{small} Problem

Theorem 2. *The algorithm described in Figure 3 gives an $O(\log|\mathcal{T}| + \log|\mathcal{S}|)$ approximation to the SG_{small} problem.*

Proof. If we purchase the resources according to Figure 3, then the total cost is equal to the objective value of the IP solution returned by Lemma 4. Now Lemma 2, Lemma 3 and Lemma 4 imply that this cost is at most $O(\log|\mathcal{T}| + \log|\mathcal{S}|)$ times the cost incurred by Opt. It remains to show that the solution

[3] Clearly, in $O(|\Theta|)$ time we can iterate over all possible resource types.

defends all the targets in $\mathcal{T}_{small}$ according to their threshold requirements. Fix some target $t \in \mathcal{T}_{small}$ for the rest of this proof. Given any purchased resource $r \in \mathcal{R}$, let $\boldsymbol{p}_r$ be its assignment vector according to Figure 3. Since the $X(\theta, \boldsymbol{p})$ values constitute a feasible solution to IP (9), we have that $\sum_{r \in \mathcal{R}} \sum_{s:\, t \in T(s)} \eta(\boldsymbol{p}_r, s) \geq \lambda_t$. Recall that $\eta(\boldsymbol{p}_r, s) = \boldsymbol{p}_r(s)\,|\mathcal{S}|^2$ and $\lambda_t = \lceil e \times q_t \times |\mathcal{S}|^2 \rceil \geq eq_t\,|\mathcal{S}|^2$. For all $r \in \mathcal{R}$, define $\phi_r(t) = \sum_{s:\, t \in T(s)} \boldsymbol{p}_r(s)$. Therefore, we get

$$\sum_{r \in \mathcal{R}} \phi_r(t) = \sum_{r \in \mathcal{R}} \sum_{s:\, t \in T(s)} \boldsymbol{p}_r(s) \geq eq_t \tag{14}$$

The probability that resource r does *not* defend target t is given by the expression $1 - \sum_{s:\, t \in T(s)} \boldsymbol{p}_r(s) = 1 - \phi_r(t)$. Since the event of assigning a resource to some (random) schedule occurs independently of other resources, the probability that *no resource* defends the target t is equal to $\prod_{r \in \mathcal{R}} (1 - \phi_r(t))$. Since $q_t \leq 1/e$,

$$\prod_{r \in \mathcal{R}} (1 - \phi_r(t)) \leq \prod_{r \in \mathcal{R}} \exp(-\phi_r(t)) = \exp\left(- \sum_{r \in \mathcal{R}} \phi_r(t) \right) \leq \exp(-eq_t)$$

Thus, the probability of defending target t is at least $1 - \exp(-eq_t) \geq q_t$. □

Remark. We note that it is possible to devise a polynomial time algorithm that gives an $O(\log |\mathcal{T}|)$ approximation to the SG_{small} problem. We have to consider an exponential sized Linear Program that is similar to IP (9), solve it approximately using an approximate separation oracle for its dual, and then directly employ randomized rounding. However, the running time of such an algorithm might become prohibitive. We omit the details due to space constraints.

4 Conclusion

We investigated the security game problem when there is an unlimited supply of resources that can be purchased at a cost. We designed an algorithm for (deterministically) purchasing some resources at minimum cost, and then randomly assigning them to schedules so that each target is defended with at least a certain probability. The algorithm is efficient and gives a logarithmic approximation.

Since this problem is a generalization of Set-Cover, we cannot get a sub-logarithmic approximation ratio. However, if each target has at most two schedules that are capable of defending it (a generalization of the Vertex-Cover problem) and resources are homogeneous, then we can get a constant factor approximation algorithm. We omit the proof due to lack of space. We leave open the questions of exploring other settings with better approximation guarantees, and investigating the fixed parameter tractability of the problem.

Acknowledgements. The authors thank Dmytro Korzhyk and Ronald Parr for several helpful discussions. This research was supported by NSF under award numbers IIS-0812113, IIS-0953756, CCF-1101659, CCF-0745761, CCF-1008065, a gift from Cisco, by Conitzer's Alfred P. Sloan Research Fellowship, and by Munagala's Alfred P. Sloan Research Fellowship.

References

1. B. An, J. Pita, E. Shieh, M. Tambe, C. Kiekintveld, and J. Marecki. GUARDS and PROTECT: next generation applications of security games. *ACM SIGecom Exchanges*, 10(1):31–34, 2011.
2. X. Chen and X. Deng. Settling the complexity of two-player Nash equilibrium. In *FOCS*, pages 261–272, 2006.
3. C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. In *STOC*, pages 71–78, 2006.
4. G. Dobson. Worst-case analysis of greedy heuristics for integer programming with nonnegative data. *Mathematics of Operations Research*, 7(4):515–531, 1982.
5. K. Etessami and M. Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM J. Comput.*, 39(6):2531–2597, 2010.
6. U. Feige. A threshold of $\ln n$ for approximating set-cover. *Journal of the ACM*, 45(4):634–652, 1998.
7. D. Hochbaum. *Approximation Algorithms for NP-hard Problems*. PWS Publishing Company, 1997.
8. C. Kiekintveld, M. Jain, J. Tsai, J. Pita, F. Ordóñez, and M. Tambe. Computing optimal randomized resource allocations for massive security games. In *AAMAS*, pages 689–696, 2009.
9. D. Korzhyk, V. Conitzer, and R. Parr. Complexity of computing optimal Stackelberg strategies in security resource allocation games. In *AAAI*, pages 805–810, 2010.
10. G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.
11. J. Pita, M. Jain, F. Ordóñez, C. Portway, M. Tambe, and C. Western. Using game theory for Los Angeles Airport security. *AI Magazine*, 30(1):43–57, 2009.
12. J. Pita, M. Jain, C. Western, C. Portway, M. Tambe, F. Ordóñez, S. Kraus, and P. Parachuri. Deployed ARMOR protection: The application of a game-theoretic model for security at the Los Angeles International Airport. In *AAMAS - Industry and Applications Track*, pages 125–132, 2008.
13. J. Tsai, S. Rathi, C. Kiekintveld, F. Ordóñez, and M. Tambe. IRIS - a tool for strategic security allocation in transportation. In *AAMAS - Industry Track*, pages 37–44, 2009.