Applied Adversarial Reasoning and Risk Modeling: Papers from the 2011 AAAI Workshop (WS-11-06)

Linear-Time Resource Allocation in Security Games with Identical Fully Protective Resources

Octavio Lerma[1], **Vladik Kreinovich**[1,2], **and Christopher Kiekintveld**[2]
[1]Computational Sciences Program
[2]Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
lolerma@episd.org, vladik@utep.edu
cdkiekintveld@utep.edu

Abstract

Game theory has become an important tools for making resource allocations decision in security domains, including critical infrastructure protection. Many of these games are formulated as Stackelberg security games. We present new analysis and algorithms for a class of Stackelberg security games with identical, fully protective defender resources. The first algorithm has worst-case complexity linear in the number of possible targets, but works only for a restricted case. The second algorithm can find and optimal resource allocation for the general case in time $O(n \cdot \log(n))$.

Introduction

Security problems typically involve making strategic resource allocation decisions in order to prevent or mitigate attacks. Game theory has been used to model decision-making in a variety of security situations, including the protection of critical infrastructure from terrorist attacks (Sandler and M. 2003; Bier 2007), computer network security (Alpcan and Basar 2003; Nguyen and Basar 2009; Srivastava et al. 2005), robot patrolling (Gatti 2008; Agmon et al. 2009; Halvorson, Conitzer, and Parr 2009), and scheduling (Roughgarden 2004). Recently, research on security games has been deployed to make real-world homeland security decision, including the ARMOR system in use at the LAX airport (Pita et al. 2008), the IRIS system used by the Federal Air Marshals Service (Tsai et al. 2009), and the GUARDS system developed for the Transportation Security Administration (Pita et al. 2011).

A key research direction has been the development of faster algorithms to scale to increasingly large and complex instances of security games (Conitzer and Sandholm 2006; Paruchuri et al. 2008; Kiekintveld et al. 2009; Jain et al. 2010). Faster algorithms that exploit the structure of security games have been key in enabling new applications of these methods. We present new algorithms for one of the most basic classes of security games: Stackelberg security games with multiple, identical defender resources. This class of games was described in (Kiekintveld et al. 2009), which also gave a polynomial-time ($O(n^2)$) algorithm for computing Stackelberg equilibrium of these games.

In this paper we present two new algorithms for Stackelberg security games with identical resources. The first solves a special case in worst-case linear time ($O(n)$), and the second solves the general case in $O(n \cdot \log(n))$. In addition to improving on the theoretical complexity of the best known methods for this class of security games, our algorithms are based on a detailed analysis of the structure of the solutions for these games, which may lead to faster algorithms or heuristics for more complex variants of security games.

Security Game Model: General Case

We adopt the general model of security games described in (Kiekintveld et al. 2009). A security game has two players, a *defender*, Θ, and an *attacker*, Ψ. There is a set of n targets $t_i \in T$ that the attacker wishes to attack and the defender wishes to protect. In our model, the attacker can choose to attack exactly one target from this set. The defender has a limited number of resources, $m < n$, that can be deployed to protect the targets. We assume throughout that these resources are identical, and that at most one resource can be used to protect each target.

If the attacker chooses to attack target t_i, we call the attack *successful* if the target is left uncovered by a defender, and *unsuccessful* if the target is covered by a defender. The defender's payoff for an uncovered attack is denoted $U^u_\Theta(t)$, and for a covered attack $U^c_\Theta(t)$. Similarly, $U^u_\Psi(t)$ and $U^c_\Psi(t)$ denote the attacker's payoffs in each case. We will make the standard assumptions for security games that $U^u_\Theta(t) < U^c_\Theta(t)$ and $U^u_\Psi(t) > U^c_\Psi(t)$ for all targets t. In other words, the attacker receives a higher payoff for attacking an undefended target than a defended one, and vice versa for the defender. Note that this does not imply that the games are zero-sum (or even strategically zero-sum).

The attacker's set of pure strategies consist of attacking each of the n targets. The defender's set of pure strategies comprise all possible ways to assign the m resources to the n targets. However, we can conveniently summarize the defender's strategy by defining the *coverage vector* which gives the probability that there is a defender resource assigned to each individual target. Let us denote these probabilities by c_i, so that $\sum\limits_{i=1}^{n} c_i = m$. The attacker's expected utility for an attack on target t_i can then be written as $(1-c_i)\cdot U^u_\Psi(t_i)+c_i\cdot U^c_\Psi(t_i)$, and similarly for the defender. We denote these expected utilities by Ω_Θ and Ω_Ψ for the defender and attacker, respectively. These expected utilities are a function of the strategies of both players. Because of our assumptions, for each target t_i, the defender's expected payoff decreases when the probability c_i of defending this target increases. We also assume that all defender resources are identical and can be deployed to any target.

We model the game as a *Stackelberg game* (von Stackelberg 1934) in which the attacker can observe the defender's strategy $(c_1,\ldots,c_n)$ before planning an attack (modeling the capability of attackers to use surveillance to learn security policies). The standard solution concept for these games is Strong Stackelberg Equilbrium (SSE) (Leitmann 1978; Basar and Olsder 1995). In an SSE, the leader first selects a mixed strategy, and then the follower chooses an optimal pure strategy in response, breaking ties in favor of the leader. This behavior can be induced by the leader selecting a strategy arbitrarily close to the equilibrium that causes the the follower to strictly prefer the desired strategy (von Stengel and Zamir 2004), but in practice we compute the limit point where ties are broken in favor of the leader.

More formally, let δ_Θ be a mixed strategy (i.e., coverage vector) for the defender. The attacker's strategy in a Stackelberg security game is a function that selects a strategy in response to every possible leader strategy: $F_\Psi : \Delta_\Theta \to \Delta_\Psi$, where Δ_Θ is the set of all possible defender strategies and Δ_Ψ is the set of all possible attacker pure strategies. A pair of strategies (δ_Θ, F_Ψ) form a *Strong Stackelberg Equilibrium* (SSE) if they satisfy the following:

1. The leader plays a best-response: $\Omega_\Theta(\delta_\Theta, F_\Psi(\delta_\Theta)) \geq \Omega_\Theta(\delta'_\Theta, F_\Psi(\delta'_\Theta))\;\forall\,\delta'_\Theta \in \Delta_\Theta$.
2. The follower play a best-response: $\Omega_\Psi(\delta_\Theta, F_\Psi(\delta_\Theta)) \geq \Omega_\Psi(\delta_\Theta, \delta_\Psi)\;\forall\,\delta_\Theta \in \Delta_\Theta, \delta_\Psi \in \Delta_\Psi$.
3. The follower breaks ties optimally for the leader: $\Omega_\Theta(\delta_\Theta, F_\Psi(\delta_\Theta)) \geq \Omega_\Theta(\delta_\Theta, \delta_\Psi)\;\forall\,\delta_\Theta \in \Delta_\Theta, \delta_\Psi \in \Delta^*_\Psi(\delta_\Theta)$, where $\Delta^*_\Psi(\delta_\Theta)$ is the set of follower best-responses, as above.

Our goal in this paper is to develop efficient algorithms for computing SSE solutions to instances of Stackelberg security games.

Case of Fully Protective Resources: Description and Analysis of the Problem

We begin by studying a practically and conceptually important restricted case of the problem, and later generalize our analysis to a broader class of Stackelberg security games. For now, we will assume that the attacker's payoff for attacking a target that is protected is 0 (that is, $U^c_\Psi(t)=0$), and that a single defender resources is "fully protective" so that it is never necessary to assign more than one resource to a single target. We will also assume that no target is covered 100% of the time, or that there is a single defender resource. We begin with a basic analysis that describes the structure of the solution.

The analysis presented in (Kiekintveld et al. 2009) showed that for any Stackelberg security game with identical, unconstrained resources that each cover a single target and a payoff structure such that $U^u_\Theta(t) < U^c_\Theta(t)$ and $U^u_\Psi(t) > U^c_\Psi(t)$ for all targets t, we can find an SSE solution without considering the defender's payoffs. To do so, we need to find a coverage vector that (1) minimizes the maximum expected payoff for the attacker and (2) maximizes the number of targets that have the same maximum payoff for the attacker. If we call the set of targets that are best responses for the attacker given some coverage vector C the *attack set*, then condition 2 states that we are looking for the largest possible attack set, given that we have already minimized the attacker payoff for targets within this set. Recall that in an SSE, that attacker will break ties in favor of the defender, so the defender will receive the best possible payoff for any target in the attack set. Therefore, we do not need to account for the defender's payoffs directly; they are maximized implicitly by minimizing the attacker's payoffs and maximizing the size of the attack set. We not that these conditions are not sufficient for SSE in cases where the payoff structure described above does not hold, or if defender resources have constraints or can cover more than one target (e.g., in cases where resources can patrol multiple targets on a tour).

We now proceed with a further analysis of the structure of this problem that will lead to a linear-time algorithm for computing an SSE for this class of games. The attacker seeks to maximize the expected value of a successful attack:

$$\arg\max_i\,(1-c_i)\cdot U^u_\Psi(t_i), \tag{1}$$

while the defender chooses a coverage vector to minimize the attacker's expected payoff. Let t_{i_o} denote the optimal target to attack, so we have for every target t_i:

$$(1-c_{i_o})\cdot U^u_\Psi(t_{i_o}) \geq (1-c_i)\cdot U^u_\Psi(t_i). \tag{2}$$

Now, assume that for some i this inequality is strict and that $c_i > 0$. In this case we could decrease c_i and increase the probability c_j for all j such that

$$(1-c_j)\cdot U^u_\Psi(t_j) = (1-c_{i_o})\cdot U^u_\Psi(t_{i_o}), \tag{3}$$

thus decreasing the expected payoff of the attacker.

Therefore, for the minimizing coverage vector, all targets can be divided into two groups:

- either the expected value for attacking the target is equal to the optimal value,
- or the expected value is less than the optimal value and the coverage probability assigned to the target is 0.

In other words, the optimal solution will have the property that the attacker's expected value for all targets with positive coverage probability is equal to some constant q:

$$(1-c_i)\cdot U^u_\Psi(t_i) = q. \tag{4}$$

For any target t_i with $c_i > 0$ we can calculate the necessary value of c_i as:

$$c_i = 1 - \frac{q}{U^u_\Psi(t_i)}. \tag{5}$$

For all other targets $U^u_\Psi(t_i) < q$, and therefore

$$1 - \frac{q}{U^u_\Psi(t_i)} < 0. \qquad (6)$$

Summarizing: once we know q, we can find all the probabilities c_i by using the formula

$$c_i = \max\left(1 - \frac{q}{U^u_\Psi(t_i)}, 0\right). \qquad (7)$$

For each target t_i, this formula requires a constant number of computational steps. Therefore, after q is computed, we can compute all the probabilities c_i in time $O(n)$.

We have shown that to find an optimal coverage vector it is sufficient to find the constant q, which can be used to directly compute the coverage probabilities for each target. This constant can be found from the condition that $\sum\limits_{i=1}^{n} c_i = m$, i.e., that

$$\sum_{i=1}^{n} \max\left(1 - \frac{q}{U^u_\Psi(t_i)}, 0\right) = m. \qquad (8)$$

The left-hand side of this equality decreases as q increases. So, if for some q, the resulting sum is smaller than m, this means that the optimal value q_o is smaller than q: $q_o < q$; similarly, if for some q, the resulting sum is larger than m, this means that the optimal value q_o is larger than q: $q_o > q$.

The structure of the optimal covering vector can be clarified if we sort the targets in order of descending attacker payoffs for successful attacks, so that:

$$U^u_\Psi(t_{(1)}) \geq \ldots \geq U^u_\Psi(t_{(n-1)}) \geq U^u_\Psi(t_{(n)}). \qquad (9)$$

We can also add $U^u_\Psi(t_{(0)}) \stackrel{\text{def}}{=} +\infty$ and $U^u_\Psi(t_{(n+1)}) \stackrel{\text{def}}{=} 0$, then

$$U^u_\Psi(t_{(0)}) \geq \ldots \geq U^u_\Psi(t_{(n)}) \geq U^u_\Psi(t_{(n+1)}). \qquad (10)$$

The values $U^u_\Psi(t_{(i)})$ divide the real line into intervals $[U^u_\Psi(t_{(i+1)}), U^u_\Psi(t_{(i)})]$, so the threshold constant q must be in one of these intervals, i.e., between $U^u_\Psi(t_{(k+1)})$ and $U^u_\Psi(t_{(k)})$ for some k. In this case, according to the above formula for c_i, all targets with a value greater than q (i.e., the targets $t_{(1)}$, $t_{(2)}$, …, $t_{(k)}$ in the above ordering) will be protected with positive probability, and all targets with value smaller than q (i.e., targets $t_{(k+1)}, t_{(k+2)}, \ldots$) are left unprotected. Let k denote the index of last target that has positive probability. Given the constraint that the coverage probabilities add to m, we can write:

$$\sum_{i=1}^{k}\left(1 - \frac{q}{U^u_\Psi(t_{(i)})}\right) = m, \qquad (11)$$

hence

$$k - m = q \cdot \sum_{i=1}^{k} \frac{1}{U^u_\Psi(t_{(i)})}, \qquad (12)$$

and

$$q = \frac{k - m}{\sum\limits_{i=1}^{k} \frac{1}{U^u_\Psi(t_{(i)})}}. \qquad (13)$$

So, instead of selecting q, we can simply select a threshold value k. Once we have found this k, we can then compute the threshold value q by using the above formula and then use this q to find the optimal coverage probabilities.

For the optimal value $k = k_o$, the corresponding value q is located in the interval $[U^u_\Psi(t_{(k+1)}), U^u_\Psi(t_{(k)})]$. If for some k, the value q computed by the above formula is smaller than $U^u_\Psi(t_{(k+1)})$, this means that we are trying to cover too few targets, with the same q, we can cover more, so the optimal value k_o should be larger: $k > k_o$.

Similarly, if for some k, the value q computed by the above formula is larger than $U^u_\Psi(t_{(k)})$, this means that we are trying to cover too many targets, so the optimal value k_o should be smaller: $k_o < k$.

Case of Fully Protective Resources: Linear-Time Algorithm

We now present a linear-time algorithm for finding optimal SSE solutions based on the analysis in the previous section. The algorithm is based on using bisection to find the right value of the constant q, which is then used to calculate the coverage for each target.

The algorithm maintains three lists of targets:

- the list T^c of the targets t_i about which we are sure that in the optimal coverage, these targets will be covered with a positive probability $c_i > 0$;
- the list T^u of the targets t_i about which we are sure that in the optimal coverage, these targets will not be covered ($c_i = 0$);
- the list $T^?$ of the targets t_i about which we have not yet found out whether they will be covered or not in the optimal coverage.

In the beginning, we set $T^c = T^u = \emptyset$ and

$$T^? = \{t_1, t_2, \ldots, t_n\}. \qquad (14)$$

At each stage, we will also update the value

$$S^c = \sum_{t_i \in T^c} \frac{1}{U^u_\Psi(t_i)}. \qquad (15)$$

In the beginning, since $T^c = \emptyset$, we take $S^c = 0$.

At each iteration, we do the following:

- First, we compute the median d of the values $U^u_\Psi(t_i)$ corresponding to all "undecided" targets $t_i \in T^?$.
- Then, by analyzing the elements of the undecided set $T^?$ one by one, we divide them into two subsets
$$T^+ = \{t_i : U^u_\Psi(t_i) \geq d\}, T^- = \{t_i : U^u_\Psi(t_i) < d\}. \qquad (16)$$
In the set T^+, we find the target t_{k^+} with the smallest value of $U^u_\Psi(t_i)$; in the set T^-, we find the target t_{k^-} with the largest value of $U^u_\Psi(t_i)$.
- We then compute
$$S^+ = \sum_{t_i \in T^+} \frac{1}{U^u_\Psi(t_i)}, \qquad (17)$$
$s^+ = S^c + S^+$, and $q = \dfrac{k - d}{s^+}$.

- If $q < U^u_\Psi(t_{k-})$, then, as we have argued in our analysis, this means that we are trying to cover too few targets, so definitely all the elements from the set T^+ should be covered. Thus, we replace T^c with $T^c \cup T^+$, $T^?$ with T^-, and S^c with s^+.
- If $q > U^u_\Psi(t_{k+})$, this means that we are trying to cover too many targets, so definitely all the elements from the set T^- should not be covered. Thus, we replace T^u with $T^u \cup T^-$ and $T^?$ with T^+ (and keep S^c unchanged).
- Finally, if $U^u_\Psi(t_{k-}) \le q \le U^u_\Psi(t_{k+})$, this means that this q is optimal.

Iterations continue until we find the optimal value q. Once we get the optimal value q, we can then find the optimal covering probabilities as $c_i = \max\left(1 - \dfrac{q}{U^u_\Psi(t_i)}, 0\right)$.

Let us prove that this algorithm indeed takes linear time. Indeed, at each iteration, we can compute the median in linear time (Cormen et al. 2009), and all other operations with the set $T^?$ also take time $\mathcal{T}$ linear in the number of elements $|T^?|$ of this set $T^?$: $\mathcal{T} \le C \cdot |T^?|$ for some C. We start with the set $T^?$ of size n. On the next iteration, we have a set of size $n/2$, then $n/4$, etc. Thus, the overall computation time is $\le C \cdot (n + n/2 + n/4 + \ldots) \le C \cdot 2n$, i.e., linear in n. This improves over the best known algorithm for this problem, ORIGAMI, which runs in time $O(n^2)$.

General Case: Analysis of the Problem

We now return to the general case, relaxing our previous assumption that the payoff for attacking a coverage target is 0. The general case can represent situations where a single resource does not completely protect a target, or where there is a benefit to the defender for capturing an attacker. In this case, the attacker seeks to maximize the expected value of an attack, taking into account both the payoff if the target is defended and the payoff when it is not defended:

$$\arg\max_\ell e_\ell(c_\ell), \tag{18}$$

where

$$e_\ell(c_\ell) \overset{\text{def}}{=} (1 - c_\ell) \cdot U^u_\Psi(t_\ell) + c_\ell \cdot U^c_\Psi(t_\ell), \tag{19}$$

while the defender chooses a coverage vector to minimize the attacker's expected payoff

$$e(c) = \max_\ell e_\ell(c_\ell). \tag{20}$$

In this case we must also consider that one or more targets may be covered 100% of the time ($c_i = 1$). For any given coverage vector we can divide the targets into three groups:

- the first group is all targets t_i for which $c_i = 1$, so the target is guarded with certainty;
- the second group is formed by targets t_j for which $0 < c_j < 1$; these targets are guarded part of the time;
- the third group is formed by targets t_k for which $c_k = 0$, so they are never guarded.

Intuitively, this division makes sense:

- the most important targets must be guarded no matter what,
- the least valuable targets will not be guarded at all if we do not have enough resources, and
- intermediate targets will be guarded with some probability.

Let us prove that this intuitive meaning is indeed true. To be more precise, we prove that in this game, there exists a minimizing vector $(c_1, \ldots, c_n)$ that has the following properties:

- The expected payoff $e_i(c_i)$ of each target t_i of the first group (with $c_i = 1$) is larger than or equal to the expected payoff $e_j(c_j)$ of each target t_j of the second group (with $0 < c_j < 1$):
$$e_i(c_i) \ge e_j(c_j). \tag{21}$$
- The expected payoff $e_j(c_j)$ of all target $t_j, t_{j'}$ from the second group (with $0 < c_j < 1$) is the same:
$$e_j(c_j) \ge e_{j'}(c_{j'}). \tag{22}$$
- The expected payoff $e_j(c_j)$ of each target t_j from the second group (with $0 < c_j < 1$) is larger than or equal to the expected payoff of each target t_k from the third group (with $c_k = 0$):
$$e_j(c_j) \ge e_k(c_k). \tag{23}$$

Intuitively, this makes sense: if the attacker's expected payoff from a target t_i that we guard absolutely is smaller than the expected payoff from some other target t_j that we guard with a certain probability, then it makes sense to switch some probability from target t_i to target t_j. In this case, the attacker's expected value for t_j decreases; for t_i it somewhat increases, but since it was smaller than for the target t_j, it remains smaller, and the maximum of these values $e_i(c_i)$ does not increase.

To prove this result more formally, let us start with any minimizing vector and show that by appropriate transformations it can be transformed into a minimizing vector with the desired properties.

First, let us show how we can satisfy the first property. For that, let us show that we can decrease the number of targets t_i for which $c_i = 1$ and for which, for some j, we have $0 < c_j < 1$ and $e_i(c_i) < e_j(c_j)$. Indeed, out of all such targets, let us pick a target for which the value $e_i(c_i)$ is the smallest, and let j be the corresponding target from the second group. Then, for some $\Delta > 0$, we replace c_i with $c'_i = c_i - \Delta$ and c_j with $c'_j = c_j + \Delta$. When Δ is small enough, we have $c'_i > 0$, $c'_j < 1$, and $e_i(c'_i)$ is still smaller than all the values $e_\ell(c_\ell)$ for which we had $e_i(c_i) < e_\ell(e_\ell)$.

Let us keep all the other probabilities the same: $e'_\ell = c_\ell$ for all $\ell \ne i, j$. This replacement does not change the sum $\sum c_i$, so while $c'_i \ge 0$ and $c'_j \le 1$, we still get a coverage vector. As we have mentioned, the expected value of a target decreases with the increase in the probability that this target will be guarded. Thus, when Δ increases, the value $e_i(c_i - \Delta)$ increases while the value $e_j(c_j + \Delta)$ decreases. So, while $e_i(c_i - \Delta) \le e_j(c_j + \Delta)$, we have $e_i(c_i) < e_i(c_i - \Delta) \le e_j(c_j + \Delta) < e_j(c_j)$.

Thus, $e_i(c'_i) < e_j(c_j) \le e(c) = \max\limits_{\ell} c_\ell(e_\ell)$ and similarly $e_j(c'_i) < e_j(c_j) \le e(c) = \max\limits_{\ell} c_\ell(e_\ell)$. For all other targets ℓ, we have $c'_\ell = c_\ell$ hence $e_\ell(c'_i) = e_\ell(c_\ell) \le e(c)$. Thus,

$$e(c') = \min\left(e_i(c'_i), e_j(c'_j), \min_{\ell \neq i,j} e_\ell(c'_\ell))\right) \le e(c). \tag{24}$$

Since the original vector c is a minimizing vector, the value $e(c)$ is the smallest possible value, we conclude that c' is also a minimizing vector.

Let us show that in the new minimizing vector, the number of targets ℓ from the first group for which the expected value is smaller than for some target from the second group is smaller than the same number computed based on the original minimizing vector. Indeed, in the new minimizing vector, the target t_i is no longer from group one, it is now from group two, so it is sufficient to check that this addition of a new group-two target does not lead to the appearance of a new "wrong-order" target of group one. Indeed, if for some target $t_{i'}$ from group one, we have $e_{i'}(c_{i'}) < e_i(c'_i)$, then we could not have $e_i(c_i) < e_{i'}(c_{i'})$ – because we selected Δ so small that all such inequalities remain. Thus, we have $e_{i'}(c_{i'}) \le e_i(c_i)$ but in this case $e_i(c_i) < e_j(c_j)$ implies that $e_{i'}(c_{i'}) < e_j(c_j)$ – and thus, $t_{i'}$ was the wrong-order target already in the original minimizing vector.

By applying this procedure again and again, we arrive at the new minimizing vector for which the number of wrong-order targets of group one is 0, i.e., in which the expected payoff for every target from group one is larger than or equal to the expected payoff for every target from group two.

Similarly, we can get a new minimizing vector in which the expected payoff for every target from group two is larger than or equal to the expected payoff of every target of group three.

Let us now show that we can arrive at the minimizing vector for which for all targets from group two, the expected payoff is the same. Let us show how an appropriate procedure can minimize the number of pairs $(t_j, t_{j'})$ of targets from group two for which $e_j(c_j) < e_{j'}(c_{j'})$. Indeed, let us sort all the corresponding values $e_j(c_j)$ into an increasing sequence, and let us take two neighboring values from this sequence. Similarly to the above case, we replace c_j with $c'_j = c_j - \Delta$ and $c_{j'}$ with $c'_{j'} = c_{j'} + \Delta$. Both expected values $e_j(c_j - \Delta)$ and $e_{j'}(c_{j'} + \Delta)$ linearly depend on Δ, so, by solving the corresponding linear equation, we can find Δ for which $e_j(c_j - \Delta) = e_{j'}(c_{j'} + \Delta)$. If this value Δ satisfies the conditions $c'_j = c_j - \Delta \ge 0$ and $c'_{j'} = c_{j'} + \Delta \le 1$, we get a new minimizing vector in which strict inequality holds for one fewer pair of targets from group two. If this value Δ does not satisfy one of these inequalities, this means that for some smaller value $\Delta' < \Delta$, we have either $c'_j = 0$ or $c'_{j'} = 1$. In both cases, the pairs stops being a wrong-order pair of targets from group two. One can check that no other wrong-order pairs appear after this transformation.

Let us now take the minimizing vector with the desired properties. In particular, this means that for all targets from group two, the attacker's expected value is the same. Let us denote this common value by q. Then, for every target t_j with $0 < c_j < 1$, we have

$$(1 - c_j) \cdot U^u_\Psi(t_j) + c_\ell \cdot U^{\mathrm{c}}_\Psi(t_j) = q. \tag{25}$$

So, we can calculate c_j as

$$c_j = \frac{U^u_\Psi(t_j) - q}{U^u_\Psi(t_j) - U^{\mathrm{c}}_\Psi(t_j)}. \tag{26}$$

For targets for which $U^u_\Psi(t_k) < q$, we have $c_k = 0$ – and the above ratio is negative. For targets for which $U^{\mathrm{c}}_\Psi(t_i) > q$, we have $c_i = 1$ – and the above ratio is larger than 1. Thus, if the ratio is smaller than 0, we take $c_i = 0$, and if the ratio is larger than 1, we take $c_i = 1$.

So, once we know q, for all targets t_i, we can find all the covering probabilities c_i by using the following formula:

$$c_i = \min\left(\max\left(\frac{U^u_\Psi(t_i) - q}{U^u_\Psi(t_i) - U^{\mathrm{c}}_\Psi(t_i)}, 0\right), 1\right). \tag{27}$$

This generalizes the formula for the fully protective case by introducing a more complicated calculation for the coverage assigned to partially covered targets, and adding an additional constraint that the coverage probability cannot exceed 1 for any target. For each target t_i, this formula still requires only a constant number of computational steps. Therefore, after q is computed, we can therefore compute all the probabilities c_i in time $O(n)$.

So, to find the optimal covering vector, it is sufficient to find the constant q. This constant can be found from the condition that $\sum\limits_{i=1}^{n} c_i = m$, i.e., that

$$\sum_{i=1}^{n} \min\left(\max\left(\frac{U^u_\Psi(t_i) - q}{U^u_\Psi(t_i) - U^{\mathrm{c}}_\Psi(t_i)}, 0\right), 1\right) = m. \tag{28}$$

The left-hand side of this equality decreases as q increases. So:

- If for some q, the resulting sum is smaller than m, this means that the optimal value q_o is smaller than q: $q_o < q$.
- Similarly, if for some q, the resulting sum is larger than m, this means that the optimal value q_o is larger than q: $q_o > q$.

Here, the target t_i is covered with probability $c_i > 0$ if and only if $q < U^u_\Psi(t_i)$, and the target t_i is covered with probability $c_i = 1$ if and only if $U^{\mathrm{c}}_\Psi(t_i) \ge q$. Thus, the above formula for determining q can be rewritten as follows:

$$k(q) + \sum_{i: U^{\mathrm{c}}_\Psi(t_i) < q \le U^u_\Psi(t_i)} \frac{U^u_\Psi(t_i) - q}{U^u_\Psi(t_i) - U^{\mathrm{c}}_\Psi(t_i)} = m, \tag{29}$$

where

$$k(q) \overset{\text{def}}{=} \#\{i : U^{\mathrm{c}}_\Psi(t_i) \ge q\}. \tag{30}$$

Thus, if we know the place of q with respect to all the values $U^u_\Psi(t_i)$ and $U^{\mathrm{c}}_\Psi(t_i)$, we can determine q by explicitly solving the above linear equation.

If we sort all $2n$ values $U^u_\Psi(t_i)$ and $U^{\mathrm{c}}_\Psi(t_i)$ into a decreasing sequence

$$z_0 = +\infty \ge z_1 \ge z_2 \ge \ldots \ge z_{2n-1} \ge z_{2n} \ge z_{2n+1} = 0, \tag{31}$$

we thus subdivide the real line into $2n+1$ zones $[z_{k+1}, z_k]$, within each of which the relation between q and the values $U^u_\Psi(t_i)$ and $U^c_\Psi(t_i)$ is fixed. Thus, within each zone, we can find the corresponding q and check whether this value is indeed within the corresponding zone. As a result, in order to find q, it is sufficient to find the corresponding value k.

Since the order is decreasing, the larger k, the smaller q, and the more targets we cover. The selection of the zone means that we select which targets we cover fully, and which targets we cover with a positive probability. Similar to the case of fully protective resources:

- If based on this selection, we need more than m resources – i.e., if the value q obtained from solving the above linear equation is smaller than all the values from this zone – this means that we are trying to cover too many targets, so we need to decrease k.
- Similarly, if it turns out that based on this selection, we need fewer than m resources – i.e., that the value q obtained from solving the above linear equation is larger than all the values from this zone – this means that we are trying to cover too few targets, so we can increase k.

Thus, we can use bisection to find the appropriate zone, and we arrive at the following algorithm.

General Case: $O(n \cdot \log(n))$ Algorithm

First, we sort all $2n$ values $U^u_\Psi(t_i)$ and $U^c_\Psi(t_i)$ into a decreasing sequence:

$$z_1 \geq z_2 \geq \ldots \geq z_{2n-1} \geq z_{2n}. \tag{32}$$

We then take $z_0 = +\infty$ and $x_{2n+1} = 0$, so that we get:

$$z_0 \geq z_1 \geq z_2 \geq \ldots \geq z_{2n-1} \geq z_{2n} \geq z_{2n+1}. \tag{33}$$

Then, we use bisection to find the value k for which $z_k \geq q \geq z_{k+1}$. At each stage of this bisection procedure, we keep two values ℓ and u such that $z_\ell \geq q \geq z_u$. In the beginning, we have $\ell = 0$ and $u = 2n+1$. At each iteration, we do the following:

- First, we compute the midpoint $d = (\ell + u)/2$.
- Then, under the assumption that $q \in [z_{d+1}, z_d]$, we compute
$$k_d = \#\{i : U^c_\Psi(t_i) \geq z_{d+1}\}, \tag{34}$$
then $d_0 = d - k_d$, and find q from the resulting linear equation
$$\sum_{i: U^c_\Psi(t_i) \leq z_d \leq z_{d+1} \leq U^u_\Psi(t_i)} \frac{U^u_\Psi(t_i) - q}{U^u_\Psi(t_i) - U^c_\Psi(t_i)} = d_0. \tag{35}$$
- If the resulting value q is smaller than z_d, then, according to our analysis, this means that the optimal k is larger than d, so we replace the original value ℓ with d.
- If the resulting value q is larger than z_{d+1}, then, according to our analysis, this means that the optimal k is smaller than d, so we replace the original value u with d.

The algorithm stops when $u = \ell + 1$, in which case we have the desired q. Based on this q, we can compute all coverage probabilities by using the above formula

$$c_i = \min\left(\max\left(\frac{U^u_\Psi(t_i) - q}{U^u_\Psi(t_i) - U^c_\Psi(t_i)}, 0\right), 1\right). \tag{36}$$

There is one more special case the must be considered to ensure that this solution is in fact a Strong Stackelberg Equilibrium. This case occurs when at least one target has coverage $c_i = 1$. In this case, we must ensure that the target that gives maximum payoff for the defender has an optimal payoff for the attacker (so far, we have considered only the payoffs for the attacker). This can be done by first finding the maximal covered payoff for the attacker $U^c_\Psi(t)$ for any target that has coverage probability 1. Denote this target by t_{max}. We then loop through each of the targets to determine whether the defender would achieve a higher payoff if the coverage probability was reduced so that the attackers expected payoff was equal to $U^c_\Psi(t_{max})$. We can compute the necessary coverage for each target using the equation:

$$c_i = \frac{U^c_\Psi(t_{max}) - U^u_\Theta(t_i)}{U^c_\Theta(t_i) - U^u_\Theta(t_i)} \tag{37}$$

If the defender's expected payoff for target t_i is greater than $U^c_\Psi(t_{max})$ given c_i, then we reduce the coverage probability to this new value c_i for target t_i. Note that this can only reduce the total coverage probability required. The additional coverage can either be left unallocated or assigned arbitrarily to any target for which the attacker has an expected payoff less than $U^c_\Psi(t_{max})$.

Let us prove that this algorithm indeed takes time

$$O(n \cdot \log(n)). \tag{38}$$

Sorting can be done in time $O(n \cdot \log(n))$ (Cormen et al. 2009). At each stage of the bisection method, we handle each target once, so each stage takes $O(n)$ computational steps. We start with an interval $[\ell, u]$ of size $2n$. At each stage, we replace it with a half-size interval $[\ell, d]$ or $[d, u]$. Thus, after the first iteration, we get an interval of size n, after the second, of size $n/2$, ..., and after k-th iteration, an interval of size $(2n)/2^k$. Thus, this procedure stops after $\log_2(2n)$ iterations. So, the overall computation time is indeed

$$O(n \cdot \log(n)) + O(n) \cdot \log(2n) = O(n \cdot \log(n)). \tag{39}$$

The final stage or analysis for the special case where at least one target coverage $c_i = 1$ requires two loop through each target. The first identifies the fully-covered target with maximum payoff for the attacker $U^c_\Psi(t_{max})$. The second calculates the required reduction in coverage probability to make the attacker indifferent between t_{max} and any other target, and replaces the coverage probability if a reduction is beneficial for the defender. Since this requires time $O(2 \cdot n)$, the overall complexity remains $O(n \cdot \log(n))$.

Conclusion

We have presented two new algorithms for a fundamental class of Stackelberg security games. These algorithms operate in linear time for a restricted case, and $O(n \cdot \log(n))$

for the general case, both improvements over the best known algorithms for this class of games. The algorithms are based on new analysis of the structure of the game-theoretic solutions of these games, which may provide insights to improve the efficiency of algorithms for additional classes of security games.

Acknowledgments

This work was partly supported by the National Center for Border Security and Immigration, by the National Science Foundation grants HRD-0734825 and DUE-0926721, and by Grant 1 T36 GM078000-01 from the National Institutes of Health.

References

Agmon, N.; Kraus, S.; Kaminka, G. A.; and Sadov, V. 2009. Adversarial uncertainty in multi-robot patrol. In *IJCAI-09*.

Alpcan, T., and Basar, T. 2003. A game theoretic approach to decision and analysis in network intrusion detection. In *Proc. of the 42nd IEEE Conference on Decision and Control*, 2595–2600.

Bagwell, K. 1995. Commitment and observability in games. *Games and Economic Behavior* 8:271–280.

Basar, T., and Olsder, G. J. 1995. *Dynamic Noncooperative Game Theory*. San Diego, CA: Academic Press, 2nd edition.

Bier, V. M. 2007. Choosing what to protect. *Risk Analysis* 27(3):607–620.

Conitzer, V., and Sandholm, T. 2006. Computing the optimal strategy to commit to. In *ACM EC-06*, 82–90.

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; and Stein, C. 2009. *Introduction to Algorithms*. Cambridge, MA: MIT Press, 3rd edition edition.

Gatti, N. 2008. Game theoretical insights in strategic patrolling: Model and algorithm in normal-form. In *ECAI-08*, 403–407.

Halvorson, E.; Conitzer, V.; and Parr, R. 2009. Multi-step multi-sensor hider-seeker games. In *IJCAI*.

Jain, M.; Kardes, E.; Kiekintveld, C.; Tambe, M.; and Ordonez, F. 2010. Security games with arbitrary schedules: A branch and price approach. In *National Conference on Artificial Intelligence (AAAI)*.

Kiekintveld, C.; Jain, M.; Tsai, J.; Pita, J.; Ordonez, F.; and Tambe, M. 2009. Computing optimal randomized resource allocations for massive security games. In *AAMAS-09*.

Leitmann, G. 1978. On generalized Stackelberg strategies. *Optimization Theory and Applications* 26(4):637–643.

Nguyen, K. C., and Basar, T. A. T. 2009. Security games with incomplete information. In *Proc. of IEEE Intl. Conf. on Communications (ICC 2009)*.

Paruchuri, P.; Pearce, J. P.; Marecki, J.; Tambe, M.; Ordonez, F.; and Kraus, S. 2008. Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *AAMAS-08*, 895–902.

Pita, J.; Jain, M.; Western, C.; Portway, C.; Tambe, M.; Ordonez, F.; Kraus, S.; and Parachuri, P. 2008. Depoloyed ARMOR protection: The application of a game-theoretic model for security at the Los Angeles International Airport. In *AAMAS-08 (Industry Track)*.

Pita, J.; Tambe, M.; Kiekintveld, C.; Cullen, S.; and Steigerwald, E. 2011. Guards - game theoretic security allocation on a national scale. In *International Conference on Autonomous Agents and Multiagent Systems*.

Roughgarden, T. 2004. Stackelberg scheduling strategies. *SIAM Journal on Computing* 33(2):332–350.

Sandholm, T.; Gilpin, A.; and Conitzer, V. 2005. Mixed-integer programming methods for finding Nash equilibria. In *AAAI-05*, 495–501.

Sandler, T., and M., D. G. A. 2003. Terrorism and game theory. *Simulation and Gaming* 34(3):319–337.

Srivastava, V.; Neel, J.; MacKenzie, A. B.; Menon, R.; Dasilva, L. A.; Hicks, J. E.; Reed, J. H.; and Gilles, R. P. 2005. Using game theory to analyze wireless ad hoc networks. *IEEE Communications Surveys and Tutuorials* 7(4).

Tsai, J.; Rathi, S.; Kiekintveld, C.; Ordóñez, F.; and Tambe, M. 2009. IRIS - A tools for strategic security allocation in transportation networks. In *AAMAS (Industry Track)*.

von Stackelberg, H. 1934. *Marktform und Gleichgewicht*. Vienna: Springer.

von Stengel, B., and Zamir, S. 2004. Leadership with commitment to mixed strategies. Technical Report LSE-CDAM-2004-01, CDAM Research Report.