Strategic Coordination of Human Patrollers and Mobile Sensors with Signaling for Security Games

Haifeng Xu, Kai Wang, Phebe Vayanos, Milind Tambe
Center for Artificial Intelligence in Society
University of Southern California, Los Angeles, CA 90007, USA
{haifengx,wang319,phebe.vayanos,tambe}@usc.edu

Abstract

Traditional security games concern the optimal randomized allocation of human patrollers, who can directly catch attackers or interdict attacks. Motivated by the emerging application of utilizing mobile sensors (e.g., UAVs) for patrolling, in this paper we propose the novel *Sensor-Empowered security Game* (SEG) model which captures the *joint* allocation of human patrollers and mobile sensors. Sensors differ from patrollers in that they cannot directly interdict attacks, but they can notify nearby patrollers (if any). Moreover, SEGs incorporate mobile sensors' natural functionality of strategic signaling. On the technical side, we first prove that solving SEGs is NP-hard even in zero-sum cases. We then develop a scalable algorithm `SEGer` based on the branch-and-price framework with two key novelties: (1) a novel MILP formulation for the slave; (2) an efficient relaxation of the problem for pruning. To further accelerate `SEGer`, we design a *faster* combinatorial algorithm for the slave problem, which is provably a constant-approximation to the slave problem in zero-sum cases and serves as a useful heuristic for general-sum SEGs. Our experiments demonstrate the significant benefit of utilizing mobile sensors.

Introduction

The past decade has seen significant interest in *security games*, which concern the allocation of limited security resources to protect critical targets from attack. This is driven in part by many real-world security applications (Tambe 2011; Yin et al. 2016; Rosenfeld and Kraus 2017; Bucarey et al. 2017). The security resources in most of these models and applications are *human patrollers*, who can directly interdict attacks. Recent advances in technology have stimulated the rapidly growing trend of utilizing automated sensors for patrolling and monitoring. Among these, Unmanned Aerial Vehicles (UAVs) – or more generally, *mobile sensors* – are perhaps the most widely used. Indeed, the UAV market is estimated at USD 13 billions in 2016 and the law enforcement/patrolling segment of the market is expected to account for the largest share (Market Report 2016). The advantage of mobile sensors is that they can automatically detect attacks with advanced image processing techniques (Ren et al. 2015; Bondi et al. 2017), thus serving as effective monitoring tools. Moreover, sensors can be more cost-effective additions than hiring new patrollers (all *sensors* in this paper will refer to mobile sensors, unless otherwise specified). However, the drawback of sensors is that they usually cannot directly interdict attacks in applications of interest in our work for law enforcement and wildlife protection – these sensors can only notify patrollers. This raises the natural question that we intend to address in this paper: how to incorporate the advantages of patrollers and sensors to improve security?

Particularly, we generalize the classical security game model and consider a defender with both patrollers and sensors. Only patrollers can directly interdict attacks. Sensors can only *detect* attacks and then notify nearby patrollers (if any) to come to intervene. So the interdiction effect of sensors relies on whether there are patrollers nearby. Moreover, we assume that sensors can strategically send signals (e.g., by making noise or shining lights) to deter attacks. Our goal is to compute the globally optimal defender policy, which includes the joint allocation of patrollers and sensors as well as the signaling schemes for sensors. Addressing this goal involves two key challenges. The first is to optimally coordinate the allocation of patrollers and sensors since sensors cannot interdict attacks independently. The second challenge is to design signaling schemes on top of the resource allocation, inducing a bi-level optimization problem.

Our Contributions. In this paper, we propose the novel Sensor-Empowered security Game (SEG) model, which naturally extends classical security games to capture the emerging application of utilizing UAVs to empower human patrolling. Our model integrates perhaps the most natural two functionalities of sensors for security, i.e., monitoring and signaling. On the technical side, we first prove that it is NP-hard to compute the optimal defender strategy even in zero-sum SEGs. We then propose `SEGer`, a scalable algorithm based on the branch and price framework with two novel ingredients: 1. a compact mixed integer linear program (MILP) formulation that exactly solves the NP-hard slave problem; 2. an efficient relaxation of the problem for branch-and-bound pruning. To further accelerate `SEGer`, we design a novel polynomial time algorithm that is provably a $\frac{1}{2}(1-\frac{1}{e})$-approximation to the slave problem in zero-sum cases while it serves as a useful heuristic for solving general-sum SEGs. Finally, we experimentally justify the advantage of utilizing sensors for security as well as the scalability of

our new algorithms.

Related Work. This work is related to several lines of research on security, and we discuss how we differ from each separately. The first line of research concerns UAV planning to optimize certain information gathering or monitoring objective (Stranders et al. 2013; Mersheeva and Friedrich 2015). These works consider (only) UAV planning and are usually in non-strategic settings. In contrast, our work concerns joint allocation of different types of resources and falls in a game-theoretic setting. Another interesting line of research studies adversarial patrolling games with alarm systems (Basilico, De Nittis, and Gatti 2017; Basilico et al. 2017), which also utilizes sensors (i.e., alarms) to assist patrollers. The sensors in all these works are *static* (staying at fixed locations) and do not strategically signal. Sensors in our model, however, can strategically signal and are *mobile*. Such mobility gives us the extra flexibility to optimize their (possibly randomized) allocation. Finally, our work is also related to the recent work on utilizing strategic signaling/deception to improve the defender utility in security games (Zhuang and Bier 2011; Xu et al. 2015; Rabinovich et al. ; Talmor and Agmon 2017; Guo et al. 2017). These works focus on security games with human patrollers, while our model optimizes the joint allocation of patrollers and signaling devices (i.e., sensors).

An Illustrative Example

To illustrate the basic idea, we start with a concrete example concerning the protection of conservation areas (Fang et al. 2016). Consider the problem where a defender needs to protect 8 conservation areas whose underlying geographic structure is captured by a cycle graph depicted in Figure 1 (e.g., they are the border areas of the park): each *vertex* represents an area. Edges indicate the adjacency relation among these areas. There is a poacher who seeks to attack one area. For simplicity, assume that these 8 areas are of equal importance. Particularly, if the poacher is caught by a patroller at any area, the defender [poacher] gets utility 1 [-1]; If the poacher successfully attacks an area, the defender [poacher] gets utility -5 [1.25]. The defender has only one patroller, who can protect any area in the graph. Since areas are symmetric, it is easy to see that the optimal patrolling strategy here simply assigns the only patroller to each area with equal probability $1/8$. As a result, the poacher attacks an arbitrary area, resulting in expected defender utility $1 \cdot \frac{1}{8} + (-5) \cdot \frac{7}{8} = -17/4$.

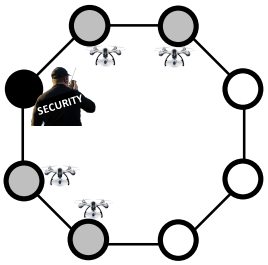


Figure 1: Cycle Graph.

Now consider that the defender is assisted by 4 UAVs (e.g., an NGO named Air Shepherd [http://airshepherd.org/] provides such UAVs for conservation). Each UAV can be assigned to patrol any area. When the poacher visits any area i, he will be caught right away if there is a patroller at i. If there is neither a patroller nor a UAV at area i, the poacher successfully attacks the target. Now if there is a UAV at i, since UAVs are usually visible by the poacher from a distance, the poacher has a chance of choosing to attack or not attack, based on his rational judgment, upon seeing the UAV. If he chooses to attack, the attack will *fail* if there is a patroller at any *neighbor* of area i, since the UAV can notify the patroller to come to catch the poacher (e.g., this is how air Shepherd operates). Otherwise, the attack succeeds (despite the presence of the UAV). The poacher can also choose to not attack, in which case both players get utility 0.

We are interested in the defender's optimal strategy for allocating these resources. By symmetry of the problem, it is natural to consider the following randomized strategy. The defender first chooses an area i uniformly at random to place the patroller, and then uses two UAVs to cover the left two neighbors of i and another two to cover the right two neighbors. The pattern is also illustrated in Figure 1 where the thick dark vertex for placing the patroller is chosen uniformly at random. Under such allocation, each vertex is assigned the patroller with probability $1/8$ and is assigned a UAV with probability $4/8$. By symmetry, the poacher still chooses an arbitrary area to visit. With probability $1/8$, the poacher will be caught by the patroller right away; with probability $3/8$, the poacher encounters neither the patroller nor the UAVs, thus will successfully conduct an attack. With the remaining $4/8$ probability, the poacher will see a UAV and need to make a choice of attacking or not attacking. It is easy to verify that conditioned on a UAV showing up at an area, with probability 0.5 there is a patroller at its neighboring area. This is because out of the four areas covered by UAVs, two of them are neighbors of the patroller-covered area. Therefore, the rational poacher will update his expected utility of committing an attack, as $(-1)\cdot 0.5 + 1.25 \cdot 0.5 = 0.125$ which is greater than the utility of not attacking. So the poacher will attack the area, resulting in expected defender utility $1 \cdot 0.5 + (-5) \cdot 0.5 = -2$. Taking expectation over all possible situations, the defender derives expected utility $1 \cdot \frac{1}{8} + (-5) \cdot \frac{3}{8} + (-2) \cdot \frac{4}{8} = -11/4$, which is an improvement over her previous utility $-17/4$.

If the defender only optimizes the allocation of these resources without extra tactics, it turns out that $-11/4$ is the maximum utility that she could possibly achieve. Interestingly, we show that the defender can further improve her utility via *strategic signaling*, which is a natural functionality of UAVs. Such improvement is possible when the poacher visits an area i covered by a UAV. In particular, let θ_{s+} [θ_{s-}] denote the random event that there is a patroller [no patroller] at some neighbor of area i. As mentioned before, conditioned on seeing a UAV at i, the poacher infers $\mathbb{P}(\theta_{s+}) = \mathbb{P}(\theta_{s-}) = 0.5$. However, the UAV will know the precise state of i through communications with the defender. The UAV could strategically signal the state of area i to the attacker with the goal of deterring his attack. This may sound counter-intuitive at first, but it turns out that strategic signaling does help. In particular, the following signaling scheme with two signals improves the defender's utility:

$$\mathbb{P}(alert|\theta_{s+}) = 1 \qquad \mathbb{P}(quiet|\theta_{s+}) = 0;$$
$$\mathbb{P}(alert|\theta_{s-}) = 0.8 \qquad \mathbb{P}(quiet|\theta_{s-}) = 0.2.$$

That is, when there is a patroller near area i (state θ_{s+}), the UAV always sends an *alert* signal; when there is no patroller near i (state θ_{s-}), 80% percent of the time the UAV still sends an *alert* signal while keeps *quiet* otherwise.

We assume that the poacher is aware of the signaling scheme and will best respond to each signal. If he receives an *alert* signal, which occurs with probability: $\mathbb{P}(alert) = \mathbb{P}(alert|\theta_{s+})\mathbb{P}(\theta_{s+}) + \mathbb{P}(alert|\theta_{s-})\mathbb{P}(\theta_{s-}) = 0.9$, the poacher infers a posterior distribution on the state by Bayes rule: $\mathbb{P}(\theta_{s+}|alert) = \frac{\mathbb{P}(alert|\theta_{s+})\mathbb{P}(\theta_{s+})}{\mathbb{P}(alert)} = \frac{5}{9}$ and $\mathbb{P}(\theta_{s-}|alert) = \frac{4}{9}$. This posterior results in expected poacher utility $\frac{5}{9}\cdot(-1) + \frac{4}{9}\cdot 1.25 = 0$, which is the same as not attacking. We assume that the poacher breaks tie in favor of the defender (see justifications later) and, in this case, chooses to not attack. This results in utility 0 for both players. On the other hand, if the poacher receives a *quiet* signal, he knows for sure that there is no patroller nearby thus chooses to attack, resulting in defender utility -5. As a result, the above signaling scheme (which occurs whenever a poacher encounters a UAV) results in defender utility $0\cdot 0.9 + (-5)\cdot 0.1 = -0.5$. Overall, the defender's expected utility is further improved to $1\cdot\frac{1}{8} + (-5)\cdot\frac{3}{8} + (-0.5)\cdot\frac{4}{8} = -2$, which is less than half of the original loss $-17/4$.

Remark. A signal takes effect only through its underlying posterior distribution over Θ_s. In the above example, the attack would not have been deterred if the UAV *always* sends an *alert* signal since in that case the poacher would ignore the signal and act based on his prior belief. However, the signals could be *deceptive* in the sense that an *alert* may be issued even when there is no patroller nearby. The poacher still prefers to not attack even he is aware of the deception!

SEGs: Sensor-Empowered Security Games

Basic Setup. Consider a security game played between a defender (she) and an attacker (he). The defender possesses k human *patrollers* and m mobile *sensors*. She aims to protect n targets, whose underlying geographic structure is captured by an undirected graph G. We use $[n]$ to denote the set of all targets, i.e., all vertices. The attacker seeks to attack one target. Let $U^{d/a}_{+/-}(i)$ denote the defender/attacker (d/a) payoff when the defender successfully protects/fails to protect $(+/-)$ the attacked target i. Assume $U^d_+(i) \geq 0 > U^d_-(i)$ and $U^a_+(i) \leq 0 < U^a_-(i)$ for any i. Sensors *cannot* directly interdict an attack, however they can inform patrollers to come when detecting the attacker at a target. Particularly, let integer $\tau > 0$ be the *intervention distance* such that a sensor-informed patroller within distance τ to the attacked target can successfully come to intervene in the attack. If there is no patroller within distance τ to the attacked target, the target is not protected despite being covered by a sensor. So a target covered by some resource (i.e., sensors) is not necessarily protected, which is a key difference between SEGs and classical security games. We assume that sensors are visible. Therefore, the attacker knows whether a target is covered by a sensor or not, upon visiting the target.

Defender's Action Space of Resource Allocation. We assume that any patroller or sensor can be assigned to cover any target on G without scheduling restrictions. Therefore, a *defender pure strategy* covers an arbitrary subset of k vertices with patrollers and another subset of m vertices with sensors. For convenience, we call both patrollers and sensors *resources*. W.l.o.g., we assume that the defender never places more than one resource at any target (otherwise, reallocating one resource to any uncovered target would only do better). Observe that targets in SEGs have 4 possible states: (1) covered by a patroller (state θ_+); (2) uncovered by any resource (state θ_-); (3) covered by a sensor and at least one patroller is within distance τ (state θ_{s+}); (4) covered by a sensor but no patroller is within distance τ (state θ_{s-}). Note that only state θ_+, θ_{s+} mean successful defense. Let $\Theta = \{\theta_+, \theta_-, \theta_{s+}, \theta_{s-}\}$ denote the set of all states. Any resource allocation uniquely determines the state for each target and vice versa. Therefore we can equivalently use a state vector $\mathbf{e} \in \Theta^n$ to denote a defender pure strategy. Let $e_i \in \Theta$ denote the state of target $i \in [n]$ and $\mathcal{E} \subseteq \Theta^n$ denote the set of defender pure strategies. A *defender mixed strategy* is a distribution over the exponentially large set $\mathcal{E}$.

Mobile Sensor Signaling. Another novel ingredient of SEGs is that they naturally integrate the sensor functionality of strategic signaling, which can be easily implemented for most sensors (e.g., UAVs). Particularly, let Σ denote the set of possible signals that a sensor could send (e.g, noise, warning lights, etc.). Let $\Theta_s = \{\theta_{s+}, \theta_{s-}\}$ denote the set of possible states when a sensor covers the target. A *signaling scheme*, w.r.t. target i, is a randomized map

$$\pi_i : \Theta_s \xrightarrow{\text{rnd}} \Sigma,$$

which is characterized by variables $\{\pi_i(e_i, \sigma_i)\}_{e_i\in\Theta_s, \sigma_i\in\Sigma}$. Here $\pi_i(e_i, \sigma_i)$ is the joint probability that target i is at state $e_i \in \Theta_s$ and signal $\sigma_i \in \Sigma$ is sent. So $\sum_{\sigma_i\in\Sigma}\pi_i(e_i, \sigma_i)$ must equal $\mathbb{P}(e_i)$, the marginal probability that target i is at state e_i. A sensor at target i first determines its state $e_i \in \Theta_s$ and then sends a signal σ_i with probability $\pi_i(e_i, \sigma_i)/\mathbb{P}(e_i)$. We assume that the defender *commits* to a signaling scheme and the rational attacker is aware of the commitment.

Upon observing signal σ_i, the attacker updates his belief on the target state: $\mathbb{P}(\theta_{s+}|\sigma_i) = \frac{\pi_i(\theta_{s+},\sigma_i)}{\pi_i(\theta_{s+},\sigma_i)+\pi_i(\theta_{s-},\sigma_i)}$ and $\mathbb{P}(\theta_{s-}|\sigma_i) = 1 - \mathbb{P}(\theta_{s+}|\sigma_i)$, and derives expected utility

$$\text{AttU}(\sigma_i) = U^a_+(i)\cdot\mathbb{P}(\theta_{s+}|\sigma_i) + U^a_-(i)\cdot\mathbb{P}(\theta_{s-}|\sigma_i).$$

The attacker will attack target i if $\text{AttU}(\sigma_i) > 0$. When $\text{AttU}(\sigma_i) < 0$, the rational attacker chooses to not attack, in which case both players get utility 0. We assume that the attacker breaks tie in favor of the defender when $\text{AttU}(\sigma_i) = 0$. This is without loss of generality because the defender can always slightly tune the probabilities to favor her preferred attacker action. The following lemma shows that the optimal signaling scheme needs not to use more than two signals.

Lemma 1. *[Adapted from (Kamenica and Gentzkow 2011)] There exists an optimal signaling scheme (w.r.t. a target) that uses at most two signals, each resulting in an attacker best response of attacking and not attacking, respectively.*

The proof of Lemma 1 tracks the following intuition: if two signals result in the same attacker best response, merging these signals into a single one would not change any

player's utility. In our previous example, an *alert* signal results in not attacking while a *quiet* signal result in attacking.

Attacker's Action Space. We assume that the defender *commits* to a mixed strategy (i.e., randomized resource allocation) and signaling schemes. The attacker is aware of the defender's commitment, and will rationally respond. In particular, the attacker first chooses a target to visit. If he observes a sensor at the target, the attacker then makes a second decision and determines to attack or not, based on the signal from the sensor. If the attacker chooses to not attack, both players get utility 0. The attacker will choose actions that maximize his utility.

Justification of Commitment and Other Assumptions.

Commitment to mixed strategies is a common assumption in security games, and has been well-justified (Tambe 2011). The commitment to signaling schemes is natural and realistic in our setting because these schemes need to be implemented as software in the sensor. Once the code is finalized and deployed, the defender is committed to use the signaling scheme prescribed by the code. We also assume that the attacker is aware of the signaling scheme and will best respond to each signal. This is because by interacting with the system, e.g., choosing to attack regardless of the signal, the attacker can gradually learn each signal's posterior which is simply a Bernoulli distribution. This is particularly true in "green security" domains which generally involve limited penalty for being caught (Carthy et al. 2016; Fang et al. 2016). Moreover, there is a community of attackers who can learn these probabilities by sharing knowledge.

Solving SEGs is Hard

We are interested in solving SEGs, by which we mean computing the *globally* optimal defender commitment consisting of the mixed strategy and signaling schemes. We first prove that solving SEGs is NP-hard even in zero-sum cases. Then we formulate the problem using the multiple-LP approach (Conitzer and Sandholm 2006). Note that all proofs in the main body are illustrated with sketches or explanations. Formal proofs can be found in the appendix.

Theorem 2. *Computing the optimal defender commitment is NP-hard even in zero-sum SEGs.*

Proof Sketch. The proof is by a reduction from the *dominating set problem*. Particularly, given any graph G with n vertices, we construct a zero-sum SEG instance with k patrollers and $m = n - k$ sensors. Let $\tau = 1$ and $U^d_+(i) = U^a_+(i) = 0, U^d_-(i) = -1 = -U^a_-(i)$ for every i. That is the defender receives utility 0 for successfully protecting a target and utility -1 for failing to protect a target. The key step of the proof is to argue that G has a dominating set of size k *if and only if* the optimal defender utility is 0 in the constructed SEG. □

A Formulation with Exponential-Size LPs

The main challenge of solving SEGs is its nature as a *bi-level* optimization problem since signaling schemes are built on top of the mixed strategy. We show that the problem can be formulated as multiple (exponential-size) LPs.

We first formulate the signaling process w.r.t. target i. For convenience, let $y_i = \mathbb{P}(e_i = \theta_{s+})$ and $z_i = \mathbb{P}(e_i = \theta_{s-})$ denote the marginal probabilities of state θ_{s+}, θ_{s-}, respectively. Thanks to Lemma 1, we can w.l.o.g. restrict to signaling schemes with two signals σ_1, σ_0 that result in the attacker best response of attacking and not attacking, respectively. Define variables $\pi^+_i = \pi_i(\theta_{s+}, \sigma_1) \in [0, y_i]$ and $\pi^-_i = \pi_i(\theta_{s-}, \sigma_1) \in [0, z_i]$. To guarantee that σ_1, σ_0 result in the desired attacker best responses, we need two constraints: $U^a_{\sigma_1}(\pi^+_i, \pi^-_i) = \pi^+_i \cdot U^a_+(i) + \pi^-_i \cdot U^a_-(i) \geq 0$ and $U^a_{\sigma_0}(\pi^+_i, \pi^-_i, y_i, z_i) = (y_i - \pi^+_i)U^a_+(i) + (z_i - \pi^-_i)U^a_-(i) \leq 0$. Under these constraints, the defender's expected utility from σ_1 is $U^d_{\sigma_1}(\pi^+_i, \pi^-_i) = \pi^+_i \cdot U^d_+(i) + \pi^-_i \cdot U^d_-(i)$. Recall that the defender utility from σ_0 is 0. Crucially, $U^a_{\sigma_1}, U^d_{\sigma_1}, U^a_{\sigma_0}$ are all *linear* functions of $\pi^+_i, \pi^-_i, y_i, z_i$.

With these representations of defender and attacker utilities from different signals, we are ready to present LPs to compute the optimal defender mixed strategy. In particular, for any fixed target t we exhibit an LP that computes the optimal defender strategy, subject to that visiting target t is the attacker's best response. Details are given in the following linear program with variables $\{p_{\mathbf{e}}\}_{\mathbf{e}\in\mathcal{E}}$ and $x_i, y_i, z_i, w_i, \pi^+_i, \pi^-_i$ for all $i \in [n]$.

$$
\begin{aligned}
\max \quad & x_t U^d_+(t) + w_t U^d_-(t) + U^d_{\sigma_1}(\pi^+_t, \pi^-_t) && \\
\text{s.t.} \quad & x_t U^a_+(t) + w_t U^a_-(t) + U^a_{\sigma_1}(\pi^+_t, \pi^-_t) \geq && \\
& \quad x_i U^a_+(i) + w_i U^a_-(i) + U^a_{\sigma_1}(\pi^+_i, \pi^-_i) && \forall\, i \neq t \\
& \textstyle\sum_{\mathbf{e}\in\mathcal{E}: e_i = \theta_+} p_{\mathbf{e}} = x_i && \forall\, i \in [n] \\
& \textstyle\sum_{\mathbf{e}\in\mathcal{E}: e_i = \theta_{s+}} p_{\mathbf{e}} = y_i && \forall\, i \in [n] \\
& \textstyle\sum_{\mathbf{e}\in\mathcal{E}: e_i = \theta_{s-}} p_{\mathbf{e}} = z_i && \forall\, i \in [n] \\
& x_i + y_i + z_i + w_i = 1 && \forall\, i \in [n] \\
& \textstyle\sum_{\mathbf{e}\in\mathcal{E}} p_{\mathbf{e}} = 1 && \\
& p_{\mathbf{e}} \geq 0 && \forall\, \mathbf{e} \in \mathcal{E} \\
& U^a_{\sigma_1}(\pi^+_i, \pi^-_i) \geq 0 && \forall\, i \in [n] \\
& U^a_{\sigma_0}(\pi^+_i, \pi^-_i, y_i, z_i) \leq 0 && \forall\, i \in [n] \\
& 0 \leq \pi^+_i \leq y_i, \qquad 0 \leq \pi^-_i \leq z_i && \forall\, i \in [n]
\end{aligned}
\tag{1}
$$

In LP (1), variable $p_{\mathbf{e}}$ is the probability of pure strategy $\mathbf{e}$ and x_i, y_i, z_i, w_i are the marginal probabilities of different states. Note that Program (1) is an LP since $U^d_{\sigma_1}, U^a_{\sigma_1}, U^a_{\sigma_0}$ are all linear functions. The last three sets of constraints guarantee that $\{\pi^+_i, \pi^-_i\}$ is a feasible signaling scheme at each target i. The first set of constraints enforce that visiting target t is an attacker best response. The remaining constraints define various marginal probabilities. It is easy to see that LP (1) computes the optimal defender commitment, subject to that visiting target t is an attacker best response.

The optimal commitment can be computed by solving LP (1) for each t and picking the solution with maximum objective. A scalable algorithm for solving SEGs is given next.

`SEGer`– A Branch and Price Approach

The challenge of solving SEGs are two-fold. First, LP (1) has exponentially many variables. Second, we have to solve LP (1) for each $t \in [n]$, which is very costly. In this section,

we propose SEGer (SEGs engine with LP relaxations) – a branch and price based algorithm – to solve SEGs. We omit the standard description of branch and price (see, e.g., (Barnhart et al. 1998)) but highlight how SEGer instantiates the two key ingredients of this framework: (a) an efficient relaxation of LP (1) for branch-and-bound pruning; (b) a column generation approach for solving LP (1). We will describe the column generation step first.

Column Generation & An MILP for the Slave

Our goal is to efficiently solve the exponential-size LP (1). The idea of column generation is to start by solving a restricted version of LP (1), where only a small subset $\mathcal{E}' \subset \mathcal{E}$ of pure strategies are considered. We then search for a pure strategy $\mathbf{e} \in \mathcal{E} \setminus \mathcal{E}'$ such that adding $\mathbf{e}$ to $\mathcal{E}'$ improves the optimal objective value. This procedure iterates until no pure strategies in $\mathcal{E} \setminus \mathcal{E}'$ can improve the objective, which means an optimal solution is found. The restricted LP (1) is called the *master*, while the problem of searching for a pure strategy $\mathbf{e} \in \mathcal{E} \setminus \mathcal{E}'$ is referred to as the *slave* problem. The slave is derived from the dual program of LP (1), particularly, from the dual constraints corresponding to primal variable $p_{\mathbf{e}}$s. We omit its textbook derivation here (see, e.g., (Tambe 2011) for details), while only directly describe the slave problem in our setting as follows.

Slave Problem: Given different weights $\alpha_i, \beta_i, \gamma_i \in \mathbb{R}$ for each i, solve the following *weight maximization problem*:

$$\text{maximize}_{\mathbf{e}\in\mathcal{E}} \sum_{i:e_i=\theta_+} \alpha_i + \sum_{i:e_i=\theta_{s+}} \beta_i + \sum_{i:e_i=\theta_{s-}} \gamma_i. \quad (2)$$

We mention that $\alpha_i, \beta_i, \gamma_i$ in the slave are the optimal dual variables for the constraints that define x_i, y_i, z_i respectively in LP (1). The slave is an interesting resource allocation problem with multiple resource types (i.e., patrollers and sensors) which affect each other. Using a reduction from dominant set, it is not difficult to prove the follows.

Lemma 3. *The slave problem is NP-hard.*

Next we propose a mixed integer linear program (MILP) formulation for the slave problem. Our idea is to use three binary vectors $\mathbf{v}^1, \mathbf{v}^2, \mathbf{v}^3 \in \{0,1\}^n$ to encode for each target whether it is in state $\theta_+, \theta_{s+}, \theta_{s-}$ respectively. For example, target i is at state θ_{s+} if and only if $v_i^2 = 1$. The main challenge then is to properly set up linear (in)equalities of these vectors to precisely capture their constraints and relations.

The capacity for each resource type results in two natural constraints: $\sum_{i\in[n]} v_i^1 \le k$ and $\sum_{i\in[n]}(v_i^2 + v_i^3) \le m$. Moreover, since at most one resource is assigned to any target, we have $v_i^1 + v_i^2 + v_i^3 \le 1$ for each $i \in [n]$. Finally, we use the set of constraints $A^\tau \cdot \mathbf{v}^1 \ge \mathbf{v}^2$ to specify which vertices could possibly have state θ_{s+} (i.e., have a patroller within distance τ). To see that this is the correct constraint, we claim that no vertex in $\mathbf{v}^1$ is within distance τ to i *if and only if* $A_i^\tau \cdot \mathbf{v}^1 = 0$ where A_i^τ is the i'th row of A^τ. This is easy to verify for $\tau = 1$ and follows by induction for general τ. It turns out that these constraints are sufficient to encode the slave problem. Details are presented in MILP (3), whose correctness is summarized in Proposition 4. Here, $\alpha = (\alpha_1, ..., \alpha_n)^\top$ (β, γ defined similarly) and $\langle \mathbf{v}^1 \cdot \alpha \rangle$ is the inner product between $\mathbf{v}^1$ and α. Matrix $A \in \{0,1\}^{n\times n}$ is the adjacency matrix of G and A^τ is the τ'th power of A.

$$\begin{array}{ll} \max & \langle \mathbf{v}^1 \cdot \alpha \rangle + \langle \mathbf{v}^2 \cdot \beta \rangle + \langle \mathbf{v}^3 \cdot \gamma \rangle \\ \text{s.t.} & \sum_{i\in[n]} v_i^1 \le k \\ & \sum_{i\in[n]}(v_i^2 + v_i^3) \le m \\ & v_i^1 + v_i^2 + v_i^3 \le 1, \qquad \text{for } i \in [n]. \\ & A^\tau \cdot \mathbf{v}^1 \ge \mathbf{v}^2 \\ & \mathbf{v}^1, \mathbf{v}^2, \mathbf{v}^3 \in \{0,1\}^n \end{array} \quad (3)$$

Proposition 4. *Let $\{\hat{\mathbf{e}}^1, \hat{\mathbf{e}}^2, \hat{\mathbf{e}}^3\}$ be an optimal solution to MILP* (3)*. Then assigning k patrollers to vertices in $\hat{\mathbf{e}}^1$ and m sensors to vertices in $\hat{\mathbf{e}}^2 + \hat{\mathbf{e}}^3$ correctly solves Slave* (2)*. Here, for vector $\mathbf{v} \in \{0,1\}^n$, we say "i is in $\mathbf{v}$"* iff $v_i = 1$.

LP Relaxation for Branch-and-Bound Pruning

Our goal of using branch-and-bound is to avoid solving LP (1) one by one for each t, which is too costly. The idea is to come up with an *efficiently-computable* upper bound of LP (1) for each t, so that once the best objective value among the solved LP (1)'s is larger than the upper bound of all the (yet) unsolved ones, we can safely claim that the current best solution is optimal without solving the remaining LPs. In this section, by properly relaxing LP (1) we obtain such an upper bound, which leads to significant speed-up in experiments.

The standard approach for finding relaxations in security games is to ignore scheduling constraints. Unfortunately, this does not work in our case since our security resources do not have scheduling constraints. The difficulty of our problem lies in characterizing marginal probabilities of different states in Θ. Our idea is to utilize the constraints in MILP (3). Observe that $\mathbf{v}^1, \mathbf{v}^2, \mathbf{v}^3$ in MILP (3) can be viewed as marginal vectors of a pure strategy for state $\theta_+, \theta_{s+}, \theta_{s-}$ respectively. Recall that $\mathbf{x}, \mathbf{y}, \mathbf{z}$ in LP (1) are the marginal vectors of a mixed strategy $\mathbf{p}$ for state $\theta_+, \theta_{s+}, \theta_{s-}$ respectively. Therefore, the $\mathbf{x}, \mathbf{y}, \mathbf{z}$ of any pure strategy must satisfy the constraints in MILP (3) by setting $\mathbf{v}^1 = \mathbf{x}$, $\mathbf{v}^2 = \mathbf{y}$, $\mathbf{v}^3 = \mathbf{z}$. By linearity, the $\mathbf{x}, \mathbf{y}, \mathbf{z}$ of any mixed strategy must also satisfy these constraints. This results in a relaxation of LP (1) by substituting the constraints in LP (1) that define x_i, y_i, z_i with the constraints of MILP (3).

Proposition 5. *The following is a valid relaxation of LP* (1)*. Moreover, this relaxation results in a linear program with polynomially number of variables and constraints.*

$$\begin{array}{l} \sum_{\mathbf{e}\in\mathcal{E}:e_i=\theta_+} p_e = x_i, \forall i \\ \sum_{\mathbf{e}\in\mathcal{E}:e_i=\theta_{s+}} p_e = y_i, \forall i \\ \sum_{\mathbf{e}\in\mathcal{E}:e_i=\theta_{s-}} p_e = z_i, \forall i \\ \sum_{\mathbf{e}\in\mathcal{E}} p_{\mathbf{e}} = 1 \\ p_{\mathbf{e}} \ge 0, \ \forall \mathbf{e} \in \mathcal{E} \end{array} \implies \begin{array}{l} \sum_{i\in[n]} x_i \le k \\ \sum_{i\in[n]}(y_i + z_i) \le m \\ x_i + y_i + z_i \le 1, \ \forall i \\ A^\tau \cdot \mathbf{x} \ge \mathbf{y} \\ \mathbf{x}, \mathbf{y}, \mathbf{z} \in [0,1]^n \end{array}$$

Relaxation: substitute left part in LP (1) with right part

A Faster Approximate Algorithm for Slave

In this section, we design a novel polynomial-time algorithm to *approximately* solve the slave problem, which can be used to accelerate SEGer. Our algorithm is provably a $\frac{1}{2}(1 - \frac{1}{e})$-approximation to the slave problem in zero-sum cases. The

approximation guarantee relies on a special property of the slave for zero-sum SEGs, stated as follows, which unfortunately is not true in general. However, the algorithm can still be used as a good *heuristic* for solving general-sum SEGs.

Lemma 6. *In zero-sum SEGs, the $\alpha_i, \beta_i, \gamma_i$ in Slave* (2) *are guaranteed to satisfy: $\alpha_i \geq \beta_i \geq \gamma_i \geq 0$ for any $i \in [n]$.*

Our algorithm originates from the following idea. That is, the slave problem can be viewed as a two-step resource allocation problem. At the first step, a vertex subset T of size at most k is chosen for allocating patrollers; At the second step, a subset $I \subseteq [n] \setminus T$ of size at most m is chosen for allocating sensors. Our key observation is that given T, the second step of choosing I is easy. Particularly, let

$$T^N = \{i \mid i \notin T \text{ but } A^{\tau}_{i,j} > 0 \text{ for some } j \in T\}$$

denote the set of all vertices that are not in T but within distance τ to some vertices in T (interpreted as *neighbors* of T). With some abuse of notions, let $T^c = [n] \setminus (T \cup T^N)$ denote the set of remaining vertices. Notice that T, T^N, T^c are mutually disjoint. The following lemma illustrates how to pick the optimal set I, given T.

Lemma 7. *Given T, the second step of the slave (i.e., picking set I) simply picks the m vertices corresponding to the largest m weights in $\{\beta_i \mid i \in T^N\} \cup \{\gamma_i \mid i \in T^c\}$.*

Lemma 7 is true because when T is given, the weight of covering target i by a sensor is determined – either β_i if $i \in T^N$ or γ_i if $i \in T^c$. Thus the main difficulty of solving the slave problem lies at the first step, i.e., to find the allocation for patrollers. For convenience, let *operator* $\Sigma^m_{\max}(W)$ denote the sum of the largest m weights in weight set W. Utilizing Lemma 7, the objective value of the slave, parameterized by set T, can be viewed as a set function of T:

$$f(T) = \textstyle\sum_{i \in T} \alpha_i + \Sigma^m_{\max}\big(\{\beta_i \mid i \in T^N\} \cup \{\gamma_i \mid i \in T^c\}\big)$$

As a result, the slave problem can be re-formulated as a *set function maximization* problem:

$$\text{Slave Reformulation:} \quad \max_{T \subset [n]: |T| \leq k} f(T)$$

The NP-hardness of the slave implies that there is unlikely a polynomial-time algorithm that maximizes $f(T)$ exactly. One natural question is whether $f(T)$ is *submodular*, since submodular maximization admits good approximation guarantees (Calinescu et al. 2011). Unfortunately, the answer turns out to be "No" (see the supplementary material for a counter example). Nevertheless, we show that maximizing $f(T)$ admits a constant approximation under conditions.

Theorem 8. *When $\alpha_i \geq \beta_i \geq \gamma_i \geq 0, \forall i \in [n]$, there is a poly-time $\frac{1}{2}(1 - \frac{1}{e})$-approximate algorithm for the slave.*

To prove Theorem 8, our key insight is that though $f(T)$ is not submodular, a carefully-crafted *variant of* $f(T)$, defined below, can be proved to be submodular. Particularly, let

$$g(T) = \textstyle\sum_{i \in T} \alpha_i + \Sigma^m_{\max}\big(\{\beta_i \mid i \in T^N \cup T\} \cup \{\gamma_i \mid i \in T^c\}\big)$$

The only difference between $f(T)$ and $g(T)$ is that the weight set in the definition of $f(T)$ [resp., $g(T)$] contains β_is for any $i \in T^N$ [resp., $i \in T^N \cup T$]. Notice that $g(T)$ can be evaluated in polynomial time for any $T \subseteq [n]$.

Our algorithm, named `TailoredGreedy` (details in Algorithm 1), runs the greedy algorithm for maximizing $g(T)$ and then uses the output to construct a solution for the slave, i.e., for maximizing $f(T)$. The remaining proof is divided into two parts. First, we prove that $g(T)$ is monotone submodular. This requires a somewhat intricate proof with careful analysis of the function. Then we show that `TailoredGreedy` yields a $\frac{1}{2}(1 - \frac{1}{e})$-approximation for the slave problem. The key step for proving this result is to establish the following relation between function $f(T)$ and $g(T)$: $f(T) \leq g(T) \leq 2f(T)$.

Algorithm 1 `TailoredGreedy`

Input: weights $\alpha_i, \beta_i, \gamma_i \in \mathbb{R}$ for any $i \in [n]$
Output: a pure strategy in $\mathcal{E}$

1: Initialization: $T = \emptyset$.
2: **for** t = 1 to k **do**
3: Compute $i^* = \arg\max_{i \in [n] - T}[g(T \cup \{i\}) - g(T)]$.
4: Add i^* to T
5: **end for**
6: **return** the pure strategy that covers vertices in T with patrollers and covers the m vertices corresponding to the largest m weights in $\{\beta_i \mid i \in T^N\} \cup \{\gamma_i \mid i \in T^c\}$ with sensors.

Experimental Results

In this section, we experimentally test our model and algorithms. All LPs and MILPs are solved by CPLEX (version 12.7.1) on a machine with an Intel core i5-7200U CPU and 11.6 GB memory. All the game payoffs are generated via the covariant game model (Nudelman et al. 2004). Particularly, let $\mu[a, b]$ denote the uniform distribution over interval $[a, b]$. For any $i \in [n]$, we generate $U^d_+(i) \sim \mu[0, 10], U^d_-(i) \sim \mu[-10, 0], U^a_+(i) = cor \cdot U^d_+(i) + (1 + cor) \cdot \mu[-10, 0]$ and $U^a_-(i) = cor \cdot U^d_-(i) + (1 + cor) \cdot \mu[0, 10]$ where $cor \in [-1, 0]$ is a parameter controlling the *correlation* between the defender and attacker payoffs. The game is zero-sum when $cor = -1$. All general-sum games are generated with $cor = -0.6$ unless otherwise stated. The graph G is generated via the Erdös – Rényi random graph model.

Sensors Improve the Defender's Utility

Figure 2 shows the comparison of the defender utility under different scenarios. All data points in Figure 2 are averaged over 30 random instances and each instance has 30 targets.

The left panel of Figure 2 compares the following scenarios. The defender has a fixed budget that equals the total cost of 7 patrollers, and the cost of a patroller may equal the cost of 3 or 5 or 7 sensors (corresponding to ratio 3, ratio 5 and ratio 7 line, respectively). The x-axis coordinate k means the defender gets k patrollers and $ratio \times (7 - k)$ sensors; y-axis is the defender utility. The figure demonstrates that a *proper combination* of patrollers and sensors results in better defender utility than just having patrollers (i.e., $k = 7$). This is the case even when the cost ratio is 3. The figure also shows that many sensors with few patrollers will not perform well

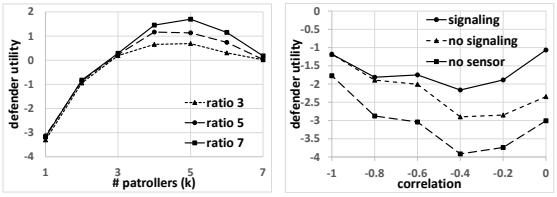


Figure 2: Utility Comparison

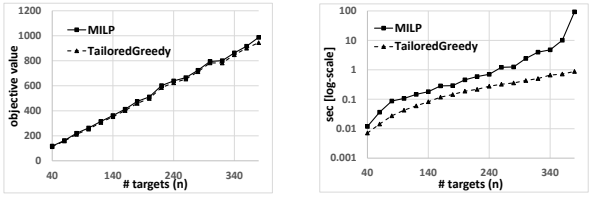
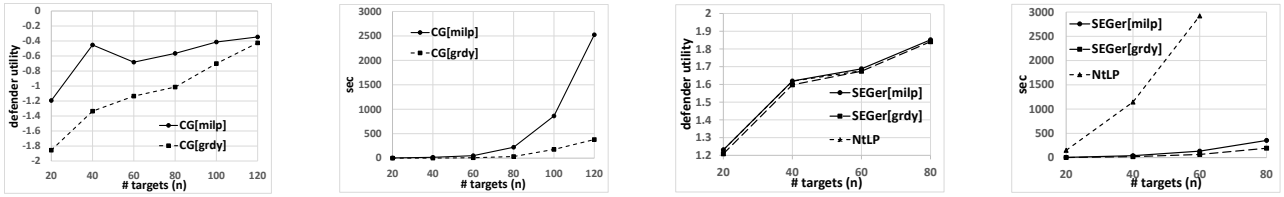


Figure 3: `TailoredGreedy` vs. MILP



(a) Zero-sum, def utility (b) Zero-sum, run time (c) General-sum, def utility (d) General-sum, run time

Figure 4: Utility comparison and scalability test of different algorithms for solving general-sum and zero-sum SEGs.

neither. Therefore, the number of patrollers and sensors need to be properly balanced in practice.

The right panel of Figure 2 compares the defender utility in three different models: 1. `signaling` – SEG model; 2. `no signaling` – SEG model but assuming sensors do not strategically signal; 3. `no sensor` – classical security games. Both `signaling` and `no signaling` have 4 patrollers and 10 sensors while `no sensor` has 6 patrollers with no sensors (i.e., cost ratio between the patroller and sensor is 5). The x-axis is the correlation parameter of the general-sum games. The graph G used in this figure is a cycle graph motivated by the protection of the border of conservation parks as in our previous illustrative example. The figure shows that `signaling` results in higher utility than `no signaling`, demonstrating the benefit of using strategic signaling in this setting. Such a benefit decreases as the game becomes closer to being zero-sum (i.e., *cor* tends to -1). This is as expected since it is well-known that signaling does not help in zero-sum cases due to its strict competition (Xu et al. 2015). Both `signaling` and `no signaling` result in a stably higher utility that `no sensor` regardless of players' payoff correlation.

`TailoredGreedy` vs. MILP

In Figure 3, we compare the performances of MILP (3) and `TailoredGreedy` on solving just the *slave* problem. Notice that running time in the right panel is in *logarithmic scale*. Each data point is an average over 15 instances with randomly generated $\alpha_i \geq \beta_i \geq \gamma_i \geq 0$ for each $i \in [n]$. Figure 3 shows that `TailoredGreedy` achieves only slightly worst objective value than MILP but is much more scalable. The scalability superiority of `TailoredGreedy` becomes particularly clear for larger instances ($n \geq$280) where MILP starts to run in exponential time while `TailoredGreedy` is a polynomial time algorithm.

Game Solving: Utility & Scalability Comparisons

Finally, we compare the performance of different algorithms in solving SEGs in Figure 4. Since zero-sum SEGs can be formulated by a single LP, which can then be solved by column generation. We compare two algorithms in this case: `CG[milp]` – column generation with MILP (3) for the slave; `CG[grdy]` –column generation with `TailoredGreedy` for the slave. Note that `CG[milp]` is optimal while `CG[grdy]` is not optimal since it uses an approximate algorithm for the slave.[1] Figure 4(b) shows that our algorithms can solve zero-sum SEGs with $80 \sim 100$ targets (depending on the algorithm) within 10 minutes. `CG[grdy]` achieves less utility than `CG[milp]`, but is more scalable (exact calculations show that `CG[grdy]` is at least 6 times faster). Interestingly, the utility gap between `CG[milp]` and `CG[grdy]` becomes smaller as n grows, while their running time gap becomes larger. This suggests that it is more desirable to use `CG[milp]` for small instances and `CG[grdy]` for large instances if some utility loss is acceptable.

For general-sum SEGs (Figures 4(c) and 4(d)), we consider three algorithms: 1. `SEGer[milp]` – `SEGer` using MILP for column generation; 2. `SEGer[grdy]` – `SEGer` using `TailoredGreedy` for column generation; 3. `NtLP` – solving LP (1) one by one for each t without branch and bound. Surprisingly, though `SEGer[grdy]` is not optimal, it achieves close-to-optimal objective value in this case and runs faster than `SEGer[milp]` (roughly half of the running time of `SEGer[milp]`). On the other hand, both `SEGer[milp]` and `SEGer[grdy]` are much more scalable than `NtLP`. In fact, the running time for solving a general-sum SEG by `SEGer[milp]` is only slightly more than the running time of solving a zero-sum SEG of the same size, which demonstrates the significant advantage of our branch and price algorithm.

[1] We also implemented the algorithm that uses `TailoredGreedy` first and then switch to MILP when `TailoredGreedy` does not improve the objective. However, this approach seems to not help in our case and results in the same running time as `CG[milp]`, thus we do not present it here.

Conclusions and Future Work

In this paper, we initiated the study of strategic coordination of human patrollers and mobile sensors. We proposed the SEG model, which integrates sensors' functionalities of monitoring and signaling into security games, and provided an algorithmic study for the model. Our work raises several opening directions for future research. One important question is to consider sensors' false positive/negative detections in the model. It is also interesting to analyze the advantages of using mobile sensors compared to static ones. Finally, our work did not consider scheduling constraints and patrol path planning for sensors and patrollers, which is an intriguing direction for future research.

Acknowledgment: This research was supported by MURI grant W911NF-17-1-0370. Haifeng Xu was supported by a Google PhD Fellowship.

References

Barnhart, C.; Johnson, E. L.; Nemhauser, G. L.; Savelsbergh, M. W.; and Vance, P. H. 1998. Branch-and-price: Column generation for solving huge integer programs. *Operations research* 46(3):316–329.

Basilico, N.; Celli, A.; De Nittis, G.; and Gatti, N. 2017. Coordinating multiple defensive resources in patrolling games with alarm systems. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS.

Basilico, N.; De Nittis, G.; and Gatti, N. 2017. Adversarial patrolling with spatially uncertain alarm signals. *Artificial Intelligence* 246:220–257.

Bondi, E.; Fang, F.; Kar, D.; Noronha, V.; Dmello, D.; Tambe, M.; Iyer, A.; and Hannaford, R. 2017. Viola: Video labeling application for securitydomains. In *Conference on Decision and Game Theory for Security (GameSec) 2017*.

Bucarey, V.; Casorrn, C.; Óscar Figueroa; Rosas, K.; Navarrete, H.; and Ordónez, F. 2017. Building real stackelberg security games for border patrols. In *Conference on Decision and Game Theory for Security (GameSec) 2017*.

Calinescu, G.; Chekuri, C.; Pál, M.; and Vondrák, J. 2011. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*.

Carthy, S. M.; Tambe, M.; Kiekintveld, C.; Gore, M. L.; and Killion, A. 2016. Preventing illegal logging: simultaneous optimization of resource teams and tactics for security. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 3880–3886. AAAI Press.

Conitzer, V., and Sandholm, T. 2006. Computing the optimal strategy to commit to. In *Proceedings of the 7th ACM conference on Electronic commerce*, 82–90. ACM.

Fang, F.; Nguyen, T. H.; Pickles, R.; Lam, W. Y.; Clements, G. R.; An, B.; Singh, A.; and Tambe, M. 2016. Deploying paws to combat poaching: Game-theoretic patrolling in areas with complex terrain (demonstration). In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, 4355–4356. AAAI Press.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, first edition edition.

Guo, Q.; An, B.; Bosansky, B.; and Kiekintveld, C. 2017. Comparing strategic secrecy and stackelberg commitment in security games. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*.

Kamenica, E., and Gentzkow, M. 2011. Bayesian persuasion. *American Economic Review* 101(6):2590–2615.

Market Report. 2016. *Unmanned Aerial Vehicle (UAV) Market, by Application - Global Forecast to 2022*.

Mersheeva, V., and Friedrich, G. 2015. Multi-uav monitoring with priorities and limited energy resources. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling*, 347–355. AAAI Press.

Nudelman, E.; Wortman, J.; Shoham, Y.; and Leyton-Brown, K. 2004. Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*, 880–887. IEEE Computer Society.

Rabinovich, Z.; Jiang, A. X.; Jain, M.; and Xu, H. Information disclosure as a means to security. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS, Istanbul, Turkey, 2015*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Rosenfeld, A., and Kraus, S. 2017. When security games hit traffic: Optimal traffic enforcement under one sided uncertainty. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*.

Stranders, R.; De Cote, E. M.; Rogers, A.; and Jennings, N. R. 2013. Near-optimal continuous patrolling with teams of mobile information gathering agents. *Artificial intelligence* 195:63–105.

Talmor, N., and Agmon, N. 2017. On the power and limitations of deception in multi-robot adversarial patrolling. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 430–436.

Tambe, M. 2011. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press.

Xu, H.; Rabinovich, Z.; Dughmi, S.; and Tambe, M. 2015. Exploring information asymmetry in two-stage security games. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*.

Yin, Y.; Vorobeychik, Y.; An, B.; and Hazon, N. 2016. Optimally protecting elections. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 538–545. AAAI Press.

Zhuang, J., and Bier, V. M. 2011. Secrecy and deception at equilibrium, with applications to anti-terrorism resource allocation. *Defence and Peace Economics* 22(1):43–61.

APPENDIX

Proof of Theorem 2

We reduce from the *dominating set problem*. A dominating set for a graph G is a subset D of vertices such that every vertex is either in D or adjacent to a vertex in D. The dominating set problem is to compute the size of a smallest dominating set for G. This problem is NP-hard even when G is a planar graph with maximum degree 3 (Garey and Johnson 1979). We now reduce an arbitrary dominating set instance to our problem.

Given any graph G with n vertices, consider a zero-sum SEG instance with k patrollers and $m = n - k$ sensors. Let $\tau = 1$ and $U^d_+(i) = U^a_+(i) = 0, U^d_-(i) = -1 = -U^a_-(i)$ for every i. That is the defender receives utility 0 for successfully protecting a target and utility -1 for failing to protect a target. We now prove that G has a dominating set of size k *if and only if* the optimal defender utility is 0 in the constructed SEG. As a result, by solving SEGs, we can solve the dominating set problem by enumerating different k's, yielding the NP-hardness of solving SEGs.

$\Rightarrow$: If G has a dominating set D of size k, we can cover the k vertices in D with patrollers and cover all the rest vertices with sensors. By definition, any vertex not in D, covered by a sensor, will be adjacent to a vertex in D therefore is successfully protected. As a result, all vertices are successfully protected and the defender receives utility 0.

$\Leftarrow$: If the defender achieves utility 0, this must imply that each target is always successfully protected, i.e., either at state s_+ or s_{s+} (signaling is not even needed in this case). Otherwise, since attack failure has cost 0 to the attacker ($U^a_+(i) = 0$), the attacker will attack a target that is protected with probability $p < 1$, which would have resulted in a negative defender utility – a contraction. This implies that any pure strategy must successfully protect every target, which means the vertices protected by the k patrollers must form a dominating set.

Proof of Lemma 3

The proof is similar to the proof of Theorem 2. In particularly, by letting $\alpha_i = \beta_i = 1, \gamma_i = 0, \tau = 1$ and $m = n - k$, it is easy to show that the graph has an independent set of size k if and only if the slave problem has optimal objective value n.

Proof of Proposition 4

We prove that feasible solutions to MILP (3) precisely encodes all pure strategies in $\mathcal{E}$, under the mapping that vertices in $\hat{\mathbf{e}}^1$ have state s_+, vertices in $\hat{\mathbf{e}}^2$ have state s_{s+} and vertices in $\hat{\mathbf{e}}^3$ have state s_{s-}. As a result, the objective of MILP (3) equals the objective of slave, yielding the desired conclusion.

First, any pure strategy in $\mathcal{E}$ must satisfy all constraints of MILP (3). To see this, we only need to argue the necessity of satisfying constraint $A^\tau \cdot \mathbf{v}^1 \geq \mathbf{v}^2$. Let A_i denote the i'th row of A. Observe that the non-zero entries in A_i specify all vertices within distance 1 from i. A standard inductive argument shows that the non-zero entries in the i'th row of A^τ, denoted by A^τ_i, are precisely all the vertices within distance τ to i. Now let $\mathbf{v}^1$ denote the subset of vertices covered by patrollers, then $A^\tau_i \cdot \mathbf{v}^1 > 0$ if and only if there is a vertex in $\mathbf{v}^1$ (i.e., covered by a patroller) that is within distance τ to i. Only such a vertex i can have $e^2_i = 1$, and this is precisely captured by $A^\tau_i \cdot \mathbf{v}^1 \geq e^2_i$ for all i (i.e., $A^\tau \cdot \mathbf{v}^1 \geq \mathbf{v}^2$).

Conversely, a similar argument shows that any feasible solution to MILP (3) corresponds to a pure strategy in $\mathcal{E}$ by assigning k patrollers to vertices in $\hat{\mathbf{e}}^1$ and m sensors to vertices in $\hat{\mathbf{e}}^2 + \hat{\mathbf{e}}^3$, concluding the proof of the proposition.

Proof of Lemma 6

The linear program for solving zero-sum SEGs can be written as follows, which is a slight modification to LP (1):

$$
\begin{array}{lll}
\max & u & \\
\text{s.t.} & u \leq x_i U^d_+(i) + w_i U^d_-(i) + U^d_\sigma(\pi^+_i, \pi^-_i) & \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}: e_i = \theta_+} p_e = x_i & \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}: e_i = \theta_{s+}} p_e = y_i & \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}: e_i = \theta_{s-}} p_e = z_i & \forall\, i \in [n] \\
& x_i + y_i + z_i + w_i = 1 & \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}} p_{\mathbf{e}} = 1 & \\
& p_{\mathbf{e}} \geq 0 & \forall\, \mathbf{e} \in \mathcal{E} \\
& U^d_\sigma(\pi^+_i, \pi^-_i) \leq 0 & \forall\, i \in [n] \\
& (y_i - \pi^+_i) U^d_+(i) + (z_i - \pi^-_i) U^d_-(i) \geq 0 & \forall\, i \in [n] \\
& 0 \leq \pi^+_i \leq y_i, \qquad 0 \leq \pi^-_i \leq z_i & \forall\, i \in [n]
\end{array}
\tag{4}
$$

We first prove a useful property of the optimal solution of LP (4). In particular, we show that there always exists an optimal solution to LP (4) that satisfies $\pi^-_i = z_i \, \forall i \in [n]$.

First, we claim that it is without loss of generality to assume that the optimal solution satisfied either $y_i = \pi^+_i$ or $z_i = \pi^-_i$. Otherwise, we can increase π^+_i by $\frac{\epsilon}{U^d_+(i)}$ and π^-_i by $\frac{\epsilon}{-U^d_-(i)}$ without violating constraints and changing the objective value. Once one of the π^+_i, π^-_i reaches its upper bound, we have $y_i = \pi^+_i$ or $z_i = \pi^-_i$ and the solution remains optimal.

Now, if $\pi^-_i = z_i$, then we are done. If $\pi^+_i = y_i$, we have $0 \leq (y_i - \pi^+_i) U^d_+(i) + (z_i - \pi^-_i) U^d_-(i) = (z_i - \pi^-_i) U^d_-(i) \leq 0$, which implies $z_i = \pi^-_i$ or $U^d_-(i) = 0$. In the later case, we can arbitrary set π^-_i to be z_i without affecting anything neither.

Therefore, adding the constraint $z_i = \pi^-_i$ will not affect the optimal value of linear program (4). Moreover, π^+_i is always non-negative at the optimal solution. So relaxing π^+_i to be a real number will not affect the optimal value. Thus, the linear program 4 is equivilent to the following linear program:

$$
\begin{array}{ll}
\max & u \\
\text{s.t.} & u \le x_i U_+^d(i) + (1 - x_i - y_i - z_i) U_-^d(i) \\
& \qquad + \pi_i^+ U_+^d(i) + z_i^- U_-^d(i) \qquad \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}: e_i = \theta_+} p_e = x_i \qquad \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}: e_i = \theta_{s+}} p_e = y_i \qquad \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}: e_i = \theta_{s-}} p_e = z_i \qquad \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}} p_{\mathbf{e}} = 1 \\
& p_{\mathbf{e}} \ge 0 \qquad \forall\, \mathbf{e} \in \mathcal{E} \\
& \pi_i^+ U_+^d(i) + z_i U_-^d(i) \le 0 \qquad \forall\, i \in [n] \\
& \pi_i^+ \le y_i \qquad \forall\, i \in [n]
\end{array} \tag{5}
$$

The dual of LP (5) is the following LP.

$$
\begin{array}{ll}
\min & \sum_{i=1}^n U_-^d(i) w_i + r \\
\text{s.t.} & r \ge \sum\limits_{i: e_i = \theta_+} \alpha_i + \sum\limits_{i: e_i = \theta_{s+}} \beta_i + \sum\limits_{i: e_i = \theta_{s-}} \gamma_i \qquad \forall\, \mathbf{e} \in \mathcal{E} \\
& \alpha_i = [U_+^d(i) - U_-^d(i)] w_i \qquad \forall\, i \in [n] \\
& \beta_i = \varphi_i - w_i U_-^d(i) \qquad \forall\, i \in [n] \\
& \gamma_i = -\delta_i U_-^d(i) \qquad \forall\, i \in [n] \\
& \sum_{\mathbf{e} \in \mathcal{E}} p_{\mathbf{e}} = 1 \\
& \varphi_i = U_+^d(i) w_i - U_+^d(i) \delta_i \qquad \forall\, i \in [n] \\
& \sum_{i=1}^n w_i = 1
\end{array} \tag{6}
$$

in which $\alpha_i, \beta_i, \gamma_i$ correspond to the constraints defining x_i, y_i, z_i respectively. Note that

$$\alpha_i = [U_+^d(i) - U_-^d(i)] w_i \ge [U_+^d(i) - U_-^d(i)] w_i - U_+^d(i)\delta_i = \beta_i$$

Also, since $\varphi_i \ge 0$ and $U_+^d(i) \ge 0$ too, so we have the implicit constraint $w_i \ge \delta_i \ge 0$. Therefore,

$$
\begin{aligned}
\beta_i &= [U_+^d(i) - U_-^d(i)] w_i - U_+^d(i)\delta_i \\
&\ge [U_+^d(i) - U_-^d(i)] \delta_i - U_+^d(i)\delta_i \\
&= -U_-^d(i)\delta_i = \gamma_i
\end{aligned}
$$

Since $\gamma_i = -U_-^d(i)\delta_i \ge 0$, this implies

$$\alpha_i \ge \beta_i \ge \gamma_i \ge 0$$

Proof of Lemma 7

This is because when T is fixed, the weight of covering any target i by a sensor has been determined – either β_i if $i \in T^N$ or γ_i if $i \in T^c$. Therefore, to maximize the total weights, we simply pick the largest m elements in $\{\beta_i \mid i \in T^N\} \cup \{\gamma_i \mid i \in T^c\}$.

Counter Example to Submodularity of $f(T)$

Recall that

$$f(T) = \textstyle\sum_{i \in T} \alpha_i + \Sigma_{\max}^m \big(\{\beta_i \mid i \in T^N\} \cup \{\gamma_i \mid i \in T^c\}\big)$$

Consider a simple line graph G with 5 vertices, as in Figure 5. Let $\tau = 1$ and $m = 2$. Moreover, $\alpha_i = \beta_i = 1$ while $\gamma_i = 0$ for all $i = 1, .., 5$.

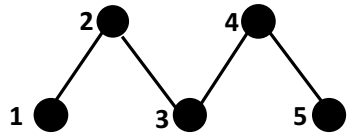


Figure 5: Graph G for the Counter Example .

Consider $S = \{2\}, T = \{2, 4\}$ and $j = 1 \notin T$. We have $f(S) = 3, f(S \cup \{j\}) = 3, f(T) = 4$ and $f(T \cup \{j\}) = 5$. Therefore,

$$f(T \cup \{j\}) - f(T) = 1 > 0 = f(S \cup \{j\}) - f(S).$$

So $f(T)$ is not submodular in T.

Proof of Theorem 8

The proof follows from the following two lemmas.

Lemma 9. *When $\alpha_i \ge \beta_i \ge \gamma_i \ge 0, \forall i \in [n]$, function $g(T)$ is nonnegative, monotone increasing and submodular.*

Proof of Lemma 9. It is easy to see that $g(T) \ge 0$ and is monotone increasing in T. We only prove its submodularity. Since $\sum_{i \in T} \alpha_i$ is a modular function of T, we only need to prove that function $f'(T) = \Sigma_{\max}^m \big(\{\beta_i \mid i \in T^N \cup T\} \cup \{\gamma_i \mid i \in T^c\}\big)$ is submodular in T. The key step is to prove that the following function is submodular:

$$W(S) = \textstyle\sum_{\max}^m (\{\beta_i | i \in S\} \cup \{\gamma_i | i \in \bar{S}\})$$

where $\beta_i \ge \gamma_i$ for all $i \in [n]$ and $\bar{S} = [n] - S$ is the complement of S. Notice that $W(T) \ne f'(T)$ (instead $W(T^N \cup T) = f'(T)$), so they are two different functions despite the similarity.

Pick any sets $S \subset T \subseteq [n]$ and $j \notin T$. Following the standard definition of submodularity, we prove the following inequality:

$$W(S \cup \{j\}) - W(S) \ge W(T \cup \{j\}) - W(T).$$

This follows a case analysis. For convenience, we will say "β_j [γ_j] contributes to $W(S)$" if β_j [γ_j] is among the largest m weights of $\{\beta_i | i \in S\} \cup \{\gamma_i | i \in \bar{S}\}$; Moreover, we denote set $S \cup \{j\}$ by S_{+j}.

- β_j contributes to $W(T_{+j})$. Then we must have that β_j also contributes to $W(S_{+j})$ since $S \subset T$. In this case, $W(S_{+j}) - W(S)$ equals β_j minus the smallest weight that contributes to $W(S)$. On the other hand, $W(T_{+j}) - W(T)$ equals β_j minus the smallest weight that contributes to $W(T)$. Since $S \subset T$, the smallest weight contributing to $W(T)$ is larger than the smallest weight contributing to $W(S)$. This implies $W(S_{+j}) - W(S) \ge W(T_{+j}) - W(T)$.
- β_j does not contribute to $W(T_{+j})$. In this case $W(T_{+j}) - W(T) = 0$ and $W(S_{+j}) - W(S) \ge 0$. Therefore, $W(S_{+j}) - W(S) \ge W(T_{+j}) - W(T)$.

As a result, $W(S)$ is submodular. We now show that $f'(T)$ is submodular by proving

$$f'(S_{+j}) - f'(S) \geq f'(T_{+j}) - f'(T)$$

for any $S \subset T \subseteq [n]$ and $j \notin T$. Let $A = T^N_{+j} \cup T_{+j} \setminus (T^N \cup T)$ and $B = S^N_{+j} \cup S_{+j} \setminus (S^N \cup S)$. Note that $A \subseteq B$ since $S \subset T$. Therefore

$$\begin{aligned} f'(S_{+j}) - f'(S) &= W(S^N_{+j} \cup S_{+j}) - W(S^N \cup S) \\ &= W(S^N \cup S \cup B) - W(S^N \cup S) \\ &\geq W(S^N \cup S \cup A) - W(S^N \cup S) \\ &\geq W(T^N \cup T \cup A) - W(T^N \cup T) \\ &= f'(T_{+j}) - f'(T), \end{aligned}$$

where the first inequality follows from monotonicity of function $W(S)$ and the second inequality follows from submodularity of $W(S)$. This proves that $f'(T)$, thus $f(T)$, is submodular. □

Lemma 10. *When $\alpha_i \geq \beta_i \geq \gamma_i \geq 0, \forall i \in [n]$, Algorithm 1 outputs a $\frac{1}{2}(1 - \frac{1}{e})$-approximation for the slave problem.*

Proof of Lemma 10. Let T^*_g and T^*_f be the optimal solution to maximizing $g(T)$ and $f(T)$ subject to $|T| \leq k$, respectively. Let $\hat{T}$ be the set generated by the greedy process (step 2 – 5) in Algorithm 1. Our goal is to prove $f(\hat{T}) \geq \frac{1}{2}(1 - \frac{1}{e}) f(T^*_f)$. The key step is to show the following relations:

$$f(T) \leq g(T) \leq 2f(T), \qquad \forall T \subseteq [n].$$

Since the $\Sigma^m_{\max}$ operator in $g(T)$ acts on a larger set than that in $f(T)$, this implies $g(T) \geq f(T)$. We now prove $g(T) \leq 2f(T)$. Since T, T^N, T^c are mutually disjoint, the weights that contribute to $f(T)$ are all indexed by different vertices. However, since $T \subseteq T^N \cup T$, there may exist vertex $i \in T$ such that both α_i and β_i contribute to $g(T)$. Let $A \subseteq T$ be all such i's. We have

$$\sum_{i \in A} \beta_i \leq \sum_{i \in A} \alpha_i \leq \sum_{i \in T} \alpha_i \leq f(T). \tag{7}$$

Moreover, if we remove the portion of $\sum_{i \in A} \beta_i$ from $g(T)$, then the left weights are all indexed by different vertices and their total weights are at most $f(T)$. That is,

$$g(T) - \sum_{i \in A} \beta_i \leq f(T) \tag{8}$$

Combining Inequalities (7) and (8) yields that $g(T) \leq f(T) + \sum_{i \in A} \beta_i \leq 2f(T)$, as desired.

By the monotone submodularity of $g(T)$ (Lemma 9), we have $g(\hat{T}) \geq (1 - \frac{1}{e}) g(T^*_g)$. Since $g(T^*_g) \geq g(T^*_f) \geq f(T^*_f)$ and $2f(\hat{T}) \geq g(\hat{T})$, this implies $f(\hat{T}) \geq \frac{1}{2}(1 - \frac{1}{e}) f(T^*_f)$. □